SANDIA REPORT
SAND2026-21635O
Printed May 20, 2026

Sandia National Laboratories

SIERRA Multimechanics Module: Aria User Manual – Version 5.30

Sierra Thermal Fluids Team

Prepared by
Sandia National Laboratories
Albuquerque, New Mexico 87185
Livermore, California 94550

Issued by Sandia National Laboratories, operated for the United States Department of Energy by National Technology & Engineering Solutions of Sandia, LLC.

NOTICE: This report was prepared as an account of work sponsored by an agency of the United States Government. Neither the United States Government, nor any agency thereof, nor any of their employees, nor any of their contractors, subcontractors, or their employees, make any warranty, express or implied, or assume any legal liability or responsibility for the accuracy, completeness, or usefulness of any information, apparatus, product, or process disclosed, or represent that its use would not infringe privately owned rights. Reference herein to any specific commercial product, process, or service by trade name, trademark, manufacturer, or otherwise, does not necessarily constitute or imply its endorsement, recommendation, or favoring by the United States Government, any agency thereof, or any of their contractors or subcontractors. The views and opinions expressed herein do not necessarily state or reflect those of the United States Government, any agency thereof, or any of their contractors.

Printed in the United States of America. This report has been reproduced directly from the best available copy.

Available to DOE and DOE contractors from

U.S. Department of Energy
Office of Scientific and Technical Information
P.O. Box 62
Oak Ridge, TN 37831

Telephone: (865) 576-8401
Facsimile: (865) 576-5728
E-Mail: reports@osti.gov
Online ordering: http://www.osti.gov/scitech

Available to the public from

U.S. Department of Commerce
National Technical Information Service
5301 Shawnee Road
Alexandria, VA 22312

Telephone: (800) 553-6847
Facsimile: (703) 605-6900
E-Mail: orders@ntis.gov
Online order: https://classic.ntis.gov/help/order-methods

Contents

1. Aria Overview . . . 9
   1.1. Introduction . . . 9
   1.2. History . . . 10
   1.3. Expression Overview . . . 10
   1.4. Unit Convention . . . 15

2. Releases & Deprecations . . . 16
   2.1. Version 5.30 (latest - 2026/05/20) . . . 16
   2.2. Version 5.28 (2026/01/23) . . . 17
   2.3. Version 5.26 (2025/09/11) . . . 18
   2.4. Version 5.24 (2025/03/19) . . . 19
   2.5. Version 5.22 (2024/10/08) . . . 21
   2.6. Version 5.20 (2024/06/17) . . . 21
   2.7. Version 5.18 (2024/02/23) . . . 22
   2.8. Version 5.16 (2023/09/26) . . . 23
   2.9. Version 5.14 (2023/05/29) . . . 24
   2.10. Version 5.12 (2023/02/09) . . . 25
   2.11. Version 5.10 (2022/10/03) . . . 26
   2.12. Version 5.8 (2022/07/11) . . . 27
   2.13. Version 5.6 (2022/04/18) . . . 28
   2.14. Version 5.4 (2022/02/14) . . . 29
   2.15. Version 5.2 (2021/11/01) . . . 30
   2.16. Version 5.0 (2021/04/26) . . . 31

3. Theory . . . 33
   3.1. Domain Definition . . . 33
   3.2. Governing Equations . . . 34
   3.3. Boundary Conditions . . . 67
   3.4. Sources . . . 70
   3.5. Interface Conditions . . . 70
   3.6. Radiation . . . 72
   3.7. Contact . . . 80
   3.8. Discretization . . . 83
   3.9. Finite Element Approximation . . . 94
   3.10. CVFEM . . . 105
   3.11. Solution Strategy . . . 107
   3.12. Coupling . . . 112
   3.13. Arbitrary Lagrangian-Eulerian . . . 115
   3.14. Teko Heuristic Multiphysics Preconditioner . . . 117

4. Simulation Setup . . . 121
   4.1. Input File Basics . . . 121

4.2. Specifying the Mesh . . . 128
4.3. Problem Specification . . . 147
4.4. Advanced Features . . . 156
4.5. User Customization . . . 292
4.6. Postprocessing . . . 337
4.7. Solving the problem . . . 361
4.8. Controlling the Simulation . . . 386
4.9. Input/Output . . . 408
4.10. Coupling . . . 426

5. Tutorials & Examples . . . 429
5.1. Sierra 100 . . . 429
5.2. Examples . . . 473

6. Running & Troubleshooting . . . 522
6.1. Running Aria . . . 522
6.2. Reading the Aria log file . . . 526
6.3. Reading the Chaparral log file . . . 541
6.4. Diagnostic Output . . . 546
6.5. Troubleshooting . . . 547
6.6. Guidelines . . . 552
6.7. FAQ . . . 556

7. Command References . . . 564
7.1. Adaptivity Settings . . . 564
7.2. Diagnostic Control . . . 573
7.3. Finite Element Model Parameters . . . 576
7.4. IO Region . . . 587
7.5. Linear Solvers . . . 594
7.6. Material Properties . . . 646
7.7. Rebalance Settings . . . 2482
7.8. Solution Control . . . 2485
7.9. Transfer . . . 2521
7.10. User Functions . . . 2537
7.11. User Fields . . . 2546
7.12. User Variables . . . 2547
7.13. Global Constants . . . 2551
7.14. Boundary Conditions . . . 2554
7.15. Equation Specification . . . 6471
7.16. Initial Conditions . . . 6496
7.17. Sources . . . 6530
7.18. Aria Region . . . 7098
7.19. Advective Bar . . . 7169
7.20. Bulk Nodes . . . 7176
7.21. CDFEM Options . . . 7183

7.22. CHEMEQ . . . 7193
7.23. Contact . . . 7212
7.24. Directed Energy . . . 7325
7.25. Element Death . . . 7358
7.26. Enclosure Radiation . . . 7361
7.27. Ray Trace Source . . . 7400
7.28. General Chemistry . . . 7406
7.29. Heat Transfer Correlation Coefficient . . . 7418
7.30. Level Set . . . 7428
7.31. Local Coordinate Systems . . . 7432
7.32. Mpmd Coupling . . . 7435
7.33. Nonlinear Solve Settings . . . 7453
7.34. Skinning . . . 7460
7.35. Segregated Solve Settings . . . 7466
7.36. Output . . . 7468
7.37. Postprocessing . . . 7506
7.38. Pressurization Model . . . 7581
7.39. Restart . . . 7586
7.40. Solution Options . . . 7600
7.41. Toggle . . . 7632
7.42. Submodel . . . 7634

8. Symbols . . . 7638

References . . . 7641

Bibliography . . . **7641**

Index . . . **7647**

List of Figures

Fig. 1.1. Example dependency graph for the calculation of the enthalpy diffusive flux. Solid lines are direct dependencies whereas dashed and dotted lines are dependencies discovered by following out-edges and recursively adding Expressions to the graph. 11
Fig. 2.1. Aria Deprecation Timeline . . . 32
Fig. 3.1. A schematic diagram of the mathematical thermal model, showing the domain Ω; the subdomains Ω_i and their interfaces $\partial\Omega_{i-j}$; and the boundary conditions on the surface Γ. . . . 33
Fig. 3.2. Schematic of 2D circular interface using a signed distance function . . . 64
Fig. 3.3. Surface Radiative Exchange . . . 73
Fig. 3.4. Radiative Transfer Circuit Model . . . 73
Fig. 3.5. Viewfactor Configuration. . . . 75

Fig. 3.6. Two arbitrary facets radiating energy to one another in a *radiation enclosure*. The energy exchanged depends on: the shape, orientation, distance, area A_i, A_j, temperatures T_i, T_j, and radiative properties of the facets ϵ_i, ϵ_j. 76
Fig. 3.7. Emissivity bands for two-surface problem. 79
Fig. 3.8. In a Discontinuous Galerkin-based contact enforcement strategy, the elements involved on the right side Γ_R, are those whose sides intersect the quadrature points of the element on the left side Γ_L. In general, the sides of elements on the right side, Γ_R^i, and Γ_R^j, may not be contiguous. 81
Fig. 3.9. Contact interface between subdomains Ω_i and Ω_j 81
Fig. 3.10. A linear finite element global basis function at a node is formed from the nodal shape functions of two neighboring elements. The global basis function for node i has support over the two elements e and $e+1$ that share the node, and takes on the value of zero everywhere else. 94
Fig. 3.11. The associated element shape functions, ψ_2^e (on local node 2 of element e), and ψ_1^{e+1} (on local node 1 of element $e+1$), are combined to define the global basis function for node i. 95
Fig. 3.12. Mapping from the master element to physical coordinates: The quadrilateral master element in (ξ, η) on the region $[-1,1] \times [-1,1]$ is mapped to its image in physical coordinates (x, y). 96
Fig. 3.13. The master tetrahedron element is mapped to its image in physical coordinates (x, y, z). 100
Fig. 3.14. The master hexahedron element in (ξ, η, ζ) on the region $[-1,1] \times [-1,1] \times [-1,1]$ is mapped to its image in physical coordinates (x, y, z). 102
Fig. 3.15. The midplane of the master triangular shell element is mapped to its image in physical coordinates (x, y, z), and the variation in the shell thickness is represented with the element shape functions. 104
Fig. 3.16. The midplane of the master quadrilateral shell element is mapped to its image in physical coordinates (x, y, z), and the variation in the shell thickness is represented with the element shape functions. 105
Fig. 3.17. Control Volume is Centered about Finite-Element Node 106
Fig. 4.1. Enumerated blocks 130
Fig. 4.2. Resulting skinned surface for all blocks in thick dashed red. Left domain shows the results of skinning all blocks, while the right shows the skinning of only `block_1` and `block_2`. 135
Fig. 4.3. Resulting intersecting skinning surface for `block_2` and `block_3` in thick dashed red 136
Fig. 4.4. Resulting skinned `block_2` including its internal faces in thick dashed red 136
Fig. 4.5. Resulting skinned exterior of `block_2` and `block_3` with shared faced removed . 137
Fig. 4.6. Skinned Shell on top face of blocks with differing recession rates 138
Fig. 4.7. Standard and Cyclic Periodicity 154
Fig. 4.8. Discretization and Associated Closed Enclosure Surface. 160
Fig. 4.9. Discretization and Associated Partial Enclosure Surface 162
Fig. 4.10. Two-Body Radiative Transfer Model. 162
Fig. 4.11. Two Surface Network Model. 162

Fig. 4.12. Partial Enclosure Radiative Transfer Model. 163
Fig. 4.13. Three Surface Network Model. 163
Fig. 4.14. Three Surface Network Model ($\varepsilon_3 = 0$ and $\varepsilon_3 = 1$). 164
Fig. 4.15. Calculation of contact distance per integration point. 184
Fig. 4.16. Contact interface and coverage mask 191
Fig. 4.17. Schematic of an advective bar 196
Fig. 4.18. Schematic of an advective bar block set up between two mesh blocks 198
Fig. 4.19. Schematic of entrance length w.r.t bar coordinates 201
Fig. 4.20. Hemispherical source geometry. 206
Fig. 4.21. Spherical source geometry. 207
Fig. 4.22. Parallelpiped source geometry. 208
Fig. 4.23. Ellipsoid source geometry. 208
Fig. 4.24. Double ellipsoid source geometry. 209
Fig. 4.25. Conical source geometry. 210
Fig. 4.26. Schematic of 1D smooth Heaviside as a function of Φ 218
Fig. 4.27. Schematic of an original BLOCK_1 mesh decomposed into separate parts by CDFEM 221
Fig. 4.28. Example domain for pressure zone setup. 282
Fig. 4.29. Time Step Selection Schematic 406
Fig. 6.1. Temperature ratio versus Fourier number for various Biot numbers 553
Fig. 6.2. Temperature ratio versus Fourier number for various Biot numbers 554
Fig. 6.3. Interpolated function and gradient for linear function 558
Fig. 6.4. Interpolated function and gradient for cubic function 558
Fig. 6.5. Convergence study domain 559
Fig. 6.6. Convergence study 559

List of Tables

Table 1.1. Commonly used values for <OPERATOR> prefix 12
Table 1.2. Valid values of the <PHASE> suffix 13
Table 1.3. Valid values of the <MATERIAL_PHASE> prefix 13
Table 1.4. Commonly used component values for <COMPONENT> suffix 13
Table 1.5. Field States 14
Table 1.6. Fundamental SI units 15
Table 3.1. Weights and coordinates for the Gauss quadrature rules on elements with linear shape functions. 98
Table 4.1. Parameter Definitions 237
Table 4.2. Feature table for CHEMEQ and General Chemistry. 251
Table 4.3. Temperature averaging methods 284
Table 8.1. List Of Symbols 7638
Table 8.2. General Notation 7639

This page intentionally left blank.

1 Aria Overview

1.1 Introduction

Note that a web version of this manual is also available.

The SIERRA Multimechanics Module: Aria, henceforth referred to as Aria for brevity, is an application implementing the finite element method (FEM) for solving systems of partial differential equations (PDEs). Foremost, Aria's development targets applications which involve incompressible flow (Navier-Stokes for $Re < 1$). However, the general design of Aria lends itself to the solution of systems of PDEs describing physical processes including energy transport, species transport with reactions, electrostatics and general transport of scalar, vector and tensor quantities in two and three dimensions both transient and direct to steady state. Moreover, different regions of the physical domain (i.e., the input mesh) may have either different materials and/or different collections of physics (viz., PDEs) defined on them. These systems of equations may be solved alone, in a segregated but coupled algorithm ("loosely coupled") or as a single, fully-coupled ("monolithic") system. Additionally, Aria can be loosely coupled to the quasi-statics solid mechanics code Adagio using the coupling application Arpeggio.

Aria is able to accommodate meshes that utilize linear and quadratic elements in two and three dimensions. In two dimensions, Aria supports quadrilateral (4 and 9 node) and triangular (3 and 6 node) elements. In three dimensions, Aria supports hexahedral (8 and 27 node) and tetrahedral (4 and 10 node) elements. Moreover, meshes may be comprised of combinations of these elements (i.e., both quadrilateral and triangular elements in two dimensions).

The physical coordinates and mesh displacements are always interpolated in accordance with the input mesh, but other solution degrees of freedom may be interpolated using a lower order basis function. For example, if the input mesh is composed of 9 node (quadratic) quadrilateral elements, then the physical coordinates and mesh displacements (if active) will be interpolated using quadratic basis functions, whereas other degrees of freedom, e.g., temperature or voltage, could use linear shape functions.

Documents containing results obtained through use of this software should cite

```
@TECHREPORT{
  SAND2026-21635O,
  author = "Sierra Thermal Fluids Team",
  title = "SIERRA Multimechanics Module: Aria User Manual - Version 5.30",
  institution = "Sandia National Laboratories",
  year = "2026",
  type = "{SAND Report}",
  number = "2026-21635O",
  address = "Albuquerque, NM and Livermore, CA"
}
```

1.2 History

In many respects, the predecessor code to Aria is the highly successful Goma code. Although none of the Goma code is included in Aria, Goma has inspired many of the algorithm, design and implementation decisions in Aria. Several of the staff members who contributed to Goma have also contributed to Aria, either as developers, users, or testers.

Like Goma, Aria's primary nonlinear algorithm is a full Newton-Raphson method with analytical sensitivities. However, Aria takes aspects of the implementation further by adding versatility in how the sensitivities are constructed and providing options for both finite-difference, and automatic differentiation sensitivities. Furthermore, Aria has a richer suite of nonlinear solver capabilities including matrix-free Newton-Krylov, methods and even loose coupling. Like its muse Goma, Aria includes a large suite of physics and capabilities that can be included in an analysis.

On December 5, 2000 Phil Sackinger initiated the Opera code which was to serve as a test bed for what would become Aria. The Opera code was a patterned after the Calore code which is a thermal analysis GFEM code, but differed in many respects. At that time, no applications had been written on top of the Sierra Framework that used multiple degrees of freedom (e.g., velocity AND pressure) or that used multiple element type (e.g., linear and quadratic interpolations). Opera was created as an exploratory code, to determine what changes would need to be made to the Framework and to experiment with the Framework in constructing such an application. During this period, the principle developers of Opera where Phil Sackinger, Sam Subia and Matt Hopkins.

On November 20, 2001 Matt Hopkins made the first commit to the Aria code base, using Opera as the starting point. And thus Aria was born.

1.3 Expression Overview

Aria makes extensive use of the Expression system. An "Expression" is a small, isolated and reusable object; each expression has dependencies among other expressions that, when automatically organized together to define operation sequence, define a directed acyclic graph (DAG). This DAG represents the original mathematical system that models a multi-physics application. A detailed description of the expression graph system is given by Notz et al. [1]. In this section, we give a brief overview.

The Expression Graph

Consider the following example of an energy diffusive flux that is comprised of the Fourier heat flux and (for a multispecies system) the species enthalpy diffusive flux ($\boldsymbol{q}$):

$$\boldsymbol{q} = -k\nabla T + \sum_{i=1}^{n_s} h_i \boldsymbol{J}_i \tag{1.1}$$

Where k is the thermal conductivity, h_i is the species enthalpy for species index i, T is the temperature and $\boldsymbol{J}_i$ is the species diffusive flux. For this example, we assume a simple form for the species diffusive flux using Fick's law:

$$\boldsymbol{J}_i = -\rho D_i \nabla Y_i \tag{1.2}$$

Once the expression dependencies for computing the diffusive flux (1.1) internally declared and resolved within Aria, a DAG is constructed to compute the desired quantity. An example of this DAG is shown in Fig. 1.1 (adapted from Notz et al. [1]).

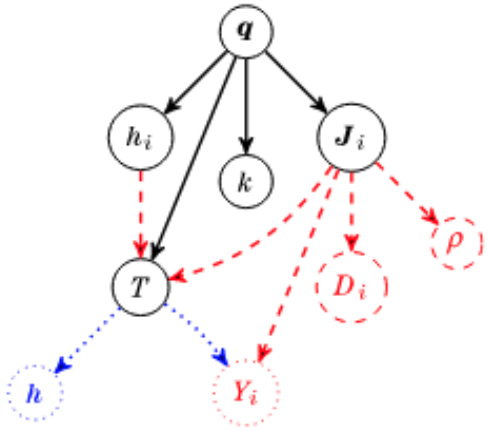


Fig. 1.1: Example dependency graph for the calculation of the enthalpy diffusive flux. Solid lines are direct dependencies whereas dashed and dotted lines are dependencies discovered by following out-edges and recursively adding Expressions to the graph.

When compared to more "traditional" programming model of multiphysics environment, the Expression system offers a more algorithmically flexible approach by deducing data dependency of various models from the assembled graph (such as the one shown in Fig. 1.1); this eliminates the need for the application programmer to specify any complex logic for specific physics models and offers a generic method for generating an assembly routine for arbitrarily complex multiphysics simulations.

General Naming Convention

Due to the dynamic nature of fields, variables and expressions in Aria a consistent naming convention must be used for sanity sake. This section describes the format of string-names of Aria Expressions. These string forms are used for input and output only; Aria has more efficient internal structures for referencing Expressions. The general format of Aria's string-based naming conventions for expression is as follows:

```
[<OPERATOR>_][<MATERIAL_PHASE>_]<NAME>[_<SUBINDEX>][_<PHASE>][_<COMPONENT>
↪]
```

In the above syntax, square quantities within square brackets are optional. Common values for the <NAME> field include degrees of freedom (such as VELOCITY, TEMPERATURE, etc.) and material properties (VISCOSITY, DENSITY, etc.). A subset of common valid values for <OPERATOR> are listed in Table Table 1.1.

Table 1.1: Commonly used values for <OPERATOR> prefix

Operator	Description
(none)	"No-Op", no-operator, Gauss point value
DT	Time derivative
GRAD	Gradient
DIV	Divergence
DET	Determinant of a second order tensor
DETJ	Determinant of the Jacobian of transformation for volume elements
SURFACE_DETJ	Determinant of the Jacobian of transformation for surface elements
REF_FRAME	"No-Op" in the undeformed reference frame
GRAD_REF_FRAME	Gradient in the undeformed reference frame
OLD	"No-Op" at previous time step
REALLY_OLD	"No-Op" at previous n-2 time step (for second order integrators)
GRAD_OLD	Gradient at the previous step
DIV_OLD	Divergence at the previous time step

The parameter <SUBINDEX> is used to designate multiple instances of a single field, and is typically used for species equations. All integer values are valid subindex values, but it's best to use values ≥ 1.

The <PHASE> field is used to delineate between two phases separated by a level set interface; some fields are present in "all phases", while others (such as material properties) depend on which phase is being referred to. Table Table 1.2 list valid values for this identifier:

Table 1.2: Valid values of the <PHASE> suffix

Phase	Description
(none)	Field is present in all phases
A	Phase A material
B	Phase B material
C	Phase C material

The <MATERIAL_PHASE> field is used to delineate between physical phases (e.g. solid, liquid, gas); some fields are present in "no material phase", while others depend on which phase is being referred to. Table Table 1.3 list valid values for this identifier:

Table 1.3: Valid values of the <MATERIAL_PHASE> prefix

Phase	Description
(none)	Field is present in all phases (NO_MATERIAL_PHASE)
LIQUID_PHASE	Liquid phase material
GAS_PHASE	Gas phase material
SOLID_PHASE	Solid phase material
WETTING_PHASE	The wetting phase (e.g. liquid, less common)
NONWETTING_PHASE	The non-wetting phase (e.g. gas, less common)

Lastly, the COMPONENT field describes a component of either a vector or tensor field; valid values are listed in Table Table 1.4

Table 1.4: Commonly used component values for <COMPONENT> suffix

Component	Description
(none)	No specified component
X	First vector component
Y	Second vector component
Z	Third vector component
XX	(1,1) 2-tensor component (default: 0)
XY	(1,2) 2-tensor component (default: 0)
XZ	(1,3) 2-tensor component (default: 0)
YX	(2,1) 2-tensor component (default: 0)
YY	(2,2) 2-tensor component (default: 0)
YZ	(2,3) 2-tensor component (default: 0)
ZX	(3,1) 2-tensor component (default: 0)
ZY	(3,2) 2-tensor component (default: 0)
ZZ	(3,3) 2-tensor component (default: 0)

Understanding these components can help the user understand how to postprocess various quantities in their models; for example, postprocessing a quantity within a liquid phase material can be done with:

```
postprocess value of expression TEMPERATURE of Li+ in LIQUID_PHASE on all_
↪blocks as foo
```

Note that in place of mangling the string, most line commands support the syntax. . .

```
[Quantity] [(Of|Species|Subindex) (SpeciesName)] [(In|Material_Phase)␣
↪(MaterialPhaseName)] [(LevelSet_Phase|LS) (A|B|C)]
```

Equation-specific Naming Convention

String names of Aria equations also have similar restrictions posed in *General Naming Convention*. These formats are used for input and output only; Aria has an internal structure for referencing specific equations. Equation names follow the below format:

```
<EQUATION_NAME>[_<SUBINDEX>][_<PHASE>][_<COMPONENT>]
```

Again, values within square brackets are optional. Some valid values for <EQUATION_NAME> include MOMENTUM, ENERGY, SPECIES, LEVEL_SET etc. The <SUBINDEX>, <PHASE> and <COMPONENT> values are the same as those described in *General Naming Convention*.

Internal Field Definition

Internally, Aria contains an internal naming scheme for fields to facilitate their algorithmic usage. In most cases this naming convention will be <usage>->field_name where <usage> categorizes its internal usage. From a user perspective the <usage> category will be solution and as an example the voltage solution is internally known as solution->voltage. Application output is managed independent of the application thus output requests must correspond to the internal code Field names rather than the Field name stated in the previous section.

Note that some fields are stated- meaning that Aria retains their value at various time points. Valid names for these states are

Table 1.5: Field States

Component	Description
NONE	Corresponds to a field without state (only latest value)
NEW/NP1	Corresponds to the field value at the NEW time
OLD/N	Corresponds to the field value at the OLD time
NM[X]	Corresponds to the field value at N-Xth old time for some integer X

1.4 Unit Convention

Aria makes no a priori specification concerning the units of each term. However, as with all engineering codes, errors associated with unit conversions are quite easy to make. To help avoid these errors, it is suggested to use fundamental SI units for all inputs (listed in Table 1.6 below).

Table 1.6: Fundamental SI units

Quantity	Name	Abbreviation
length	meter (metre)	m
mass	kilogram	kg
time	second	s
electric current	ampere	A
thermodynamic temperature	kelvin	K
amount of substance	kmole	kmol
luminous intensity	candela	cd

This ensures consistency across all calculations (including the mesh dimension). If another unit system (e.g. FPS) is used, care must be taken to ensure that it is dimensionally consistent so that

$$1\ [\text{mass}] \times (1\ [\text{length}])^2\ /\ (1\ [\text{time}])^3 = 1\ [\text{power}]$$

For example if ft is used as the [length] scale, s is used as the [time] scale, and lb_m is used as the [mass] scale, then the [power] unit must be $lb_\text{m} \cdot ft^2/s^3$ *NOT* horsepower.

It is sometimes convenient to have access to quantities in different units. Rather than convert the entire input deck over, it is often sufficient (and is suggested) to simply provide the converted expression:

```
# Scope: Sierra
begin universal aria expressions
   # Convert temperature in Kelvin to Celcius:
   User Expression = scalar_string_function f = "temperature - 273.15"␣
↪user_tag = tempC
end universal aria expressions
```

Similarly if the quantity is required only in the output file, a postprocessor can be used:

```
# Scope: Sierra > Procedure > Aria Region
# Convert a length from meters to feet:
postprocess value of function "length*3.28084" on all_blocks as lengthFt
```

See *User Customization* and *Postprocessing* for more information on creating custom expressions and postprocessors respectively.

In some situations, with just one or two driving forces playing a role in a calculation, nondimensionalization of the equations can lead to a simplification of the problem statement (and

to increased solution robustness due to proper scaling of the terms in the equations). However, in complicated cases with multiple competing forces and rate constants, sticking to unit systems to specify all quantities frequently leads to less errors in engineering calculations, and also leads to the ability to incorporate third party library packages for specification of source terms and transport properties which necessarily presume to employ units systems in their application programming interfaces (API).

Attention: When only temperature difference is required and not absolute temperature, Celcius can be used in place of Kelvin (or similarly Fahrenheit can be used in place of Rankine). Special care must be taken when using a relative temperature. Common use cases that require absolute temperature include:

- Chemistry
- Radiation
- Ideal Gas models

2 Releases & Deprecations

This section provides a high-level overview of changes associated with each release. This overview includes new features, bug fixes, deprecations, etc. A specific release can be used by loading the *associated module*.

2.1 Version 5.30 (latest - 2026/05/20)

New Features

- Improved support for mass balance tied pressure contact model so that it now works with decomposing ablation models.
- The new `PENALIZED_NORMAL_DISPLACEMENT` mesh flux boundary condition was added, providing an alternate approach to the rotated kinematic BC with no displacement that uses a penalized approach instead of rotation. Similarly, the new `SPECIFIED_NORMAL_VELOCITY` mesh flux boundary condition was added as an option for specifying mesh normal velocity without a rotated BC.
- Add a new automatic enclosure shell skinning capability to allow enclosure meshes to be decoupled from the surrounding mesh, enabling adaptivity on the volume mesh without requiring viewfactors to be recalculated.
- The `CHARGE_DENSITY` equation is now no longer a beta feature, and has testing and some flexibility for diffusive flux models.
- Enabled toggling for all flux boundary condition types, not just energy.

- The Sierra transfer now includes an optional coordinate transformation that allows transfer between 2D and 3D meshes.
- Improved the Neohookean mesh stress model to better handle element collapse numerically
- Added a new `REMOVE_OLD_MESH_STRESS` model for use in material-removal mesh deformation problems where maintaining a stress-free state is important. This can only be used along with the Neohookean model currently.

Deprecations

- The `CURRENT_EXCHANGE_DENSITY` model is deprecated, use `EXCHANGE_CURRENT_DENSITY`

Bug Fixes

- Fixed parser bug where nonlinear settings specified at the region scope would be ignored depending on the region ordering + equation system usage in the input deck. Region scope nonlinear settings are now respected irrespective of input deck ordering.
- Fixed data probe bug wherein nodal probes could pick up block restrictions from location probes leading to incorrect results.
- Fixed the behavior of surface-driven mesh adaptivity to allow mesh refinement of contact interfaces.
- Fixed a bug when using a rotated kinematic boundary condition along with the line search option that would result in non-convergence.
- Fixed a bug when using scalar string functions in universal expressions that would result in a substantial increase in memory use.
- Fixed a numerical stability issue in the Butler Volmer limiter terms, improving nonlinear convergence behavior in the limited flux regime.

2.2 Version 5.28 (2026/01/23)

New Features

- Added a bulk node generalized natural convection BC for the porous enthalpy equation.
- Added a volume weighted predictor corrector error for improved timestep sizes when using adaptivity and other mesh modification methods. To enable this, put `predictor-corrector weighting = volume_weighted` in the transient time block in your input deck.
- Added new time-interpolation mode for MPMD coupling with SPARC when Aria is sub-stepping in time

- Extended the `arialog_parse` script to work on log files containing solution control-driven nonlinear iteration loops
- Added new configurations to the `teko_multiphysics` preset solver for automatically handling mesh displacement equations in ablation problems
- Added new configurations to the `teko_multiphysics` preset solver for automatically splitting bulk nodes into a separate sub-block, improving performance and memory use with ILU preconditioners on the thermal solve
- Memory usage improvements in `teko_multiphysics` preset solver: Reduce GMRES restart size from 300 to 30 (540 vector allocations saved) and avoid storing memory for adaptive subblock solvers that are no longer in use

Bug Fixes

- Fixed bug in tftk_adapt where the log file would output the wrong element and node count before/after refinement
- Fixed a bug where tftk_adapt would throw an unhelpful error when doing partial refinement on hex/quad meshes
- Fixed a bug where the predictor corrector error was computed incorrectly for cylindrical coordinate systems
- Fixed an issue when trying to apply contact enforcement to "sub-indexed" equations and disallowed using integers as species names
- Fixed a contact enforcement bug when applying contact for multiple phases between a multiphase block and a single phase block
- Fixed the mass-preserving density model to allow density-dependent properties to be used on sidesets

2.3 Version 5.26 (2025/09/11)

New Features

- Added multiphase support to the models bulk mass density = [porous_density, mass_fraction_porous_density] and transported enthalpy = porous to ensure pore volume is split accordingly to an optional saturation. Previously, these models assumed a single phase occupying the pore volume.
- Data probes now have an optional `points_outside_tol` argument to specify their behavior when they are outside any elements by more than their specified or default search tolerance. The default behavior is to abort, but they can instead be set to a specified value (nan, zero, or a large number).
- Added material phase sum models to support single DOF porous enthalpy equation wherein phases are assumed to be in thermal equilibrium.

- Viewfactors are now only recalculated if the enclosure geometry has changed - enable this by setting VIEWFACTOR UPDATE = ONLY_IF_GEOMETRY_CHANGED in the enclosure definition

Bug Fixes

- Fixed bugs in the two-axis and three-axis cases for rotated bcs. The two-axis bug (impacts edges) violated rotation invariance. The root issue was that for certain rotations / choice of basis the K-means clustering used to cluster the normals would assign a single cluster to all normals, precluding the permutation search from finding the correct optimal rotation angle for the two-axis case. The three-axis bug (impacting corners) was due to using a non-orthonormal basis when the projection of the linear system onto the new basis relies on an orthonormal basis.
- Fixing the above three-axis rotated bc bug, exposed a rotated bc wedge corner bug wherein two opposing faces ended up contributing to the same residual component, leading to an unconstrained residual component / unintended mesh motion. This is now fixed.
- Fixed mesh boundary flux bug specific to the new porous relative mass flux formulation first introduced in 5.24 release. The presence of overlapping sidesets, skinned surfaces from auto contact, and interior sidesets previously would adversely impact the automatic mesh boundary flux computed when using the relative mass flux formulation (e.g., double counting contributions potentially). The boundary mesh flux is now computed on a skinned surface of the block as opposed to any surface touching the block, avoiding the previous issues.
- Fixed contact bug wherein a) if two contact enforcements were specified for the same pair of surfaces and b) these enforcements were applied across two different contact definitions, then each contact enforcement would be double counted.
- Fix bug with rotated BC where svd algorithm did not converge to correct basis
- Fixed postprocessor bug causing lumped terms to be ignored in postprocessing if non-lumped terms are also present and would error if there were multiple lumped terms

2.4 Version 5.24 (2025/03/19)

New Features

- Added MUSCL material model for stabilizing advection in porous flow problems for use with CVFEM.
- Added relative mass flux porous flow option to stabilize advection for problems with mesh motion, currently only stabilized with CVFEM.

- Linear system re-initialization no longer happens if contact mapping is either unchanged or a subset of the prior mapping. This can substantially improve performance for some configurations on the GPU in particular.
- CHEMEQ models can now be deactivated using user-defined conditions instead of just temperature and temperature rate of change. This feature uses user-defined functions from the material definition to define a more flexible deactivation condition that the user may choose - for example based on reaction rates or other custom conditions evaluated per block.
- The gradient of user fields is now calculated automatically, so the gradient of a user field named `myfield` could be used elsewhere in Aria as `grad_myfield`.
- Added diagnostic mode for MPMD SPARC/Aria coupling - activating it with `--tftklog transfer` will output detailed information about where the mesh gap exceeds a threshold gap size based on mesh size. The user can also set a maximum gap size with `MAXIMUM TRANSFER CLIPPING DISTANCE = X` to enable an error for mismatch over that length.
- Added a *non-dimensional residual scaling option* based on the diagonal inverse of the Jacobian and maximum absolute values of the degrees of freedom.
- Disabled specifying both advection (ADV) and diffusion (DIFF) terms for non-species mass balance in order to prevent double counting of the Darcy mass flux. Either ADV or DIFF models can be used but not both.
- Add automated `teko_multiphysics` preset solver based on heuristic methods described in *Teko Heuristic Multiphysics Preconditioner*.
- Add contact search result caching to avoid un-needed linear system re-initialization calls.
- Integrate against new Trilinos version with Kokkos 4.5, leading to as much as a 2.8x speedup in MueLu solve times.

Bug Fixes

- Fixed bugs with the "per model" postprocessor related to postprocessing fluxes on enclosures with bulk nodes and using per-model postprocessed quantities in other places (e.g. in string functions in other postprocessors).
- Fixed bug with "using P0" postprocess where if multiple models were present (e.g. model A, model B), only contributions from the first model provided were used.
- Fixed bugs with the "ideal_gas" density model - was incorrectly using SPECIES instead of SPECIES_FRACTION to calculate mixture molecular weight, and had a sensitivity error.
- Fixed sensitivity error in the gradient of the foam_time_temp density model.
- Fixed issue where block names with "." and chemeq models were creating global variable names that were not allowed (replaced "." with "_" in the global variable

name).

- Fixed a bug with the ray tracer flux model "leaking" through to the wrong side on double-sided shells.
- Fixed a bug where transition layer elements were not being properly refined in simulations with dynamic mesh adaptivity.
- Resolve bug in Teko setup for solving linear systems with constraints.
- Fixed bug with residual based post processor for CVFEM which excluded sub control volume kernel contributions to the residual.

2.5 Version 5.22 (2024/10/08)

New Features

- QUAD4/HEX8/BEAM2 elements are now supported for uniform refinement in Aria using the Krino refinement library
- Added support for interface boundary conditions for the momentum equation
- Globals can now be used directly in string functions e.g. "density * my_global.global"
- Officially removed support of Encore for use in Aria input decks

Bug Fixes

- Fixed the diagnostic control block to correctly toggle some command line debugging features in tftk products
- Updated the formula for Arrhenius_Carreau_Viscosity model

2.6 Version 5.20 (2024/06/17)

New Features

- Can now use globals in string functions without using a global user expression. e.g. `density = scalar_string_function f = "temperature/my_gvar.global"`
- Linear Solver Improvements:
  - Improvements to Teko linear solvers and documentation. Please consult *Road-map for Constructing Teko Solvers* for additional information.
  - Two new MueLu preset solvers: `thermal_multigrid` and `high_aspect_continuity`.
- The MPMD coupling strategy has now been generalized for flexibility:

  - These changes are now tracked by a coupling version, which will ensure backwards compatibility with older versions of coupled apps
  - Support for *send field* and *receive field* relaxation has been added for the implicit coupling.
  - A *–tftklog coupling_output* command line switch has been added to help investigate issues with convergence in implicit couplings by outputting intermediate iterations to the results files.
- Clamping anti-windup has been added to the *PID Controller* which matches the behavior of Matlab Simulink. A conversion script has also been provided for matching Labview (see *discussion* for caveats).
- Advective bars can now be coupled to CVFEM porous enthalpy equation.

Bug Fixes

- Using a `tensor_string_function` for intrinsic permeability now automatically includes the `intrinsic_permeability_scaling` term, which is included in other intrinsic permeability models but was missing from the string function models. Also added scaling to the `Young and Todd` model and enforced consistent subindex and phase rules (the intrinsic permeability scaling is expected to be in the `SOLID_PHASE` and can use a species subindex to be applied to a per-species intrinsic permeability).
- Postprocessed string functions that use globals will now use the up-to-date global value, whereas previously the string function would have used the previous time-steps global value.

Change in Behaviors

- By default, the preset linear solvers will use the automatic convergence tolerance, see *Tolerance Settings*.
- Gradient of a vector field now returns the gradient of each vector component stored row wise. Indexes should be flipped when referring to the corresponding tensor components in string functions (e.g., T_XY, the partial derivative of the y component with respect to x, is now accessed via T_YX).

2.7 Version 5.18 (2024/02/23)

New Features

- A skinned set of shell elements can now be generated from a source surface can be created as a *skinned shell*. This can be useful in MPMD coupling applications with surfaces deforming at differing rates.
- Added new enclosure error and warning tolerance settings to check for enclosure rowsum errors. Prior releases would not error on bad rowsum errors by default, but in

5.18 there is now a default error tolerance above which row sum errors will trigger a fatal error. Check the *enclosure radiation commands* for the appropriate syntax.

- **Ray Tracer:**
  - Added optional modular block definition for ray tracer flux model
  - Expanded ray tracer capability to enable modeling of banded radiation
- **Adaptivity:**
  - Small change to enable adaptivity to be compatible with element fields (for chemeq)
  - Added timers to sierra log output for various adaptivity operations
  - Automatically un-refine elements on blocks that do not have a marker rule, to allow mesh to fully un-refine

Upcoming Deprecations (will warn in 5.18)

- Using the `skinned_blocks` location in Calore-style BCs to automatically skin blocks is deprecated. The desired skinned surface can be explicitly created as a *skinned surface*.

2.8 Version 5.16 (2023/09/26)

New Features

- Add ability to specify arbitrary *skinned surfaces* that can be used in postprocessing, BCs, etc.
- Expanded plugin compatibility to include flux BCs, initial conditions, and a few previously missing material properties.
- Off-diagonal terms are now optional for tensor string functions, consistent with the behavior of other tensor types. If omitted, they default to 0.
- Added the ability to define and use tensor mobility terms with the chemical potential transport model for species diffusion.
- Added `all_bulk_elements` alias that can be used for BCs, postprocessors, etc in the same way as all_surfaces and all_blocks.
- Added the ability to *offset DOFs* in contact enforcements. This is typically useful in tandem with *periodic contact* or for cases where a prescribed jump across a contact interface is desired.

Upcoming Deprecations (will warn in 5.16)

- Coupling with Nalu is deprecated. Switch to Sierra/PMR for MPMD participating media radiation solves.

- Specifying `skinned_blocks` to automatically create skinned block surfaces is deprecated. Refer to the *skinned surfaces* section for updated syntax.

New Deprecations (will error in 5.16)

- The `List Mesh Groups` command has been removed. It has not had any effect for several releases, so no mitigation is needed.
- The legacy line commands for specifying contact material models have been deprecated in favor of the their generic material syntax equivalents. See *Contact Materials* for more details on specifying these commands. Note that the deprecation messages should also offer exact conversions to the new syntax.
- The `Using data specification` command for flux BCs is deprecated - use the required parameters directly.
- Specifying VOLUME or MASS postprocessors on a surface is deprecated. Switch to AREA if you need the area of a surface.

2.9 Version 5.14 (2023/05/29)

New Features

- Contact between faces that are no longer geometrically in contact is now supported. An example of this is applying contact to a periodic domain. See *Periodic Contact* for information on how to use this feature.
- Contact enforcements can now be used to impose a prescribed offset across a contact interface. This can be useful in both standard contact problems and for vector/tensor DOFs in *Periodic Contact* problems. See *Offset Quantities* for information on how to use this feature.
- *User fields* can now be smoothed after transfer by specifying a number of iterations in the `USER FIELD` line.

Upcoming Deprecations (will warn in 5.14)

- The legacy line commands for specifying contact material models have been deprecated in favor of the their generic material syntax equivalents. See *Contact Materials* for more details on specifying these commands. Note that the deprecation messages should also offer exact conversions to the new syntax.
- The `Using data specification` command for flux BCs is deprecated - use the required parameters directly.

New Deprecations (will error in 5.14)

- Coupling with Scefire is deprecated - use Sierra/PMR instead.

2.10 Version 5.12 (2023/02/09)

New Features

- Expression-based reduction operations can now be performed on enclosure radiation face fields (i.e., `rad_flux`, `irradiance`, and `radiosity`) for banded and non-banded models
- Dash enclosures can now be used with skinned surface spanning multiple energy equation types (e.g., `ENERGY`, `POROUS_ENTHALPY`)
- Dash enclosures can now be specified using a mix of blocks to skin and surfaces to include, either of which may correspond to mesh groups or assemblies
- Contact enforcements can now be applied to *periodic* surfaces. Note that periodicity is currently only supported for scalar enforcements.

Bug Fixes

- Postprocessing residual-based flux fields on Q2 elements gave incorrect "checkered" results due to using nodal areas computed from reducing determinant values at the integration points. The nodal area is now computed in a manner that is consistent with the residual i.e., $A_i = \int \phi_i dA$. Line commands affected include `postprocess flux_scalar_field` and `postprocess flux_vector_field`.

Upcoming Deprecations (will warn in 5.12)

- The `Using data specification` command for flux BCs is deprecated - use the required parameters directly.
- Coupling with Scefire is deprecated - use *Nalu* instead.

New Deprecations (will error in 5.12)

- Encore functions. See *Legacy Postprocessor Conversion* for help on conversion/available tools.
- Legacy aria post-processors. See *Legacy Postprocessor Conversion* for help on conversion/available tools.
- Migration away from "magic parameter strings" requires updating user plugin expressions with the following changes or they will not compile
  1. Change constructor to take `const PrsrParamMap& params` instead of `const String& params`
  2. Do not pass a params string to the base Expression, FAD_Expression, or User_Equation_System_Utility class
  3. Retrieve parameters using one of the following commands:

```
get_required_parameter(params, "ParamName", value,␣
↪ParamCtx{this});
get_optional_parameter(params, "ParamName", value);

const auto value = get_required_parameter<double>(params,
↪"ParamName", ParamCtx{this});
const auto value = get_optional_parameter_or_
↪default(params, "ParamName", defaultValue);

// returns std::optional
const auto value = get_optional_parameter<double>(params,
↪"ParamName");
```

2.11 Version 5.10 (2022/10/03)

New Features

- Improved predictor-corrector behavior for problems with multiple equation systems. Previously all DOF fields would be predicted prior to the start of the time step, which would lead to earlier equation systems using predicted field values for the DOFs from later equation systems. Now Aria will wait to compute the predicted DOF fields until immediately prior to the solve of the corresponding equation system.
- Support use of cyclic/recursive string functions in postprocessors
- Added capability to transfer from a shell element field to a face field when using an input region.

Bug Fixes

- Fixed line search residual to follow the convention of being reported at the start of the step. Previously, a) residuals could appear to falsely stagnate when the correction tolerance criteria caused a skip in the line search and b) an extra nonlinear iteration would occur when used in combination with the nonlinear residual ratio criteria.

Upcoming Deprecations (will warn in 5.10)

- Encore functions. See *Legacy Postprocessor Conversion* for help on conversion/available tools.
- Legacy aria post-processors. See *Legacy Postprocessor Conversion* for help on conversion/available tools.
- Migration away from "magic parameter strings" requires updating user plugin expressions with the following changes or they will not compile

1. Change constructor to take `const PrsrParamMap& params` instead of `const String& params`
2. Do not pass a params string to the base Expression, FAD_Expression, or User_Equation_System_Utility class
3. Retrieve parameters using one of the following commands:

```
get_required_parameter(params, "ParamName", value,␣
↪ParamCtx{this});
get_optional_parameter(params, "ParamName", value);

const auto value = get_required_parameter<double>(params,
↪"ParamName", ParamCtx{this});
const auto value = get_optional_parameter_or_
↪default(params, "ParamName", defaultValue);

// returns std::optional
const auto value = get_optional_parameter<double>(params,
↪"ParamName");
```

New Deprecations (will error in 5.10)

- The use of `postprocess FLUX_TYPE for EQ` is no longer supported, use `postprocess FLUX_TYPE of equation EQ` instead to better distinguish between postprocessors that operate on equations vs expressions.
- Several unused ChemEq solver parameters have been deprecated. These commands have been unused and ignored for several releases.
- The EXP_DECAY density model has been deprecated. Use a scalar string function as a replacement.
- The output from the `Time Derivative = Calculated` expression has been changed from `ddt_something` to `dt_something`. Both outputs will be produced in 5.8 but in 5.10 only the `dt_` output will be present.

2.12 Version 5.8 (2022/07/11)

New Features

- Assemblies are now supported in Aria
- Using the previous guess during radiosity solves should result in improved performance; see the user manual for more details
- Added thermal expansion term in Coussy porosity model

- Added ability to use postprocessed fields in boundary conditions across equation systems

Bug Fixes

- Fix bug in line search that ignored other nonlinear acceptance criteria
- Bug fix for NAN in Bingham WLF viscosity model

Upcoming Deprecations (will warn in 5.8)

- The use of `postprocess FLUX_TYPE for EQ` is no longer supported, use `postprocess FLUX_TYPE of equation EQ` instead to better distinguish between postprocessors that operate on equations vs expressions.
- Several unused ChemEq solver parameters have been deprecated. These commands have been unused and ignored for several releases.
- The EXP_DECAY density model has been deprecated. Use a scalar string function as a replacement.
- The output from the `Time Derivative = Calculated` expression has been changed from `ddt_something` to `dt_something`. Both outputs will be produced in 5.8 but in 5.10 only the `dt_` output will be present.

2.13 Version 5.6 (2022/04/18)

New Features

- Added support for variable solidus and liquidus temperatures in melting sources and electrochemistry.
- Added support for multi-dimensional structured tables to generic models
- Added support for universal aria expressions
- Added time step (`dt`) as a standard global variable, available to use in global functions.
- Improved the Generalized_Rad and Generalized_Nat_Conv models to allow in-line specification of constants or string functions for h and Tref.
- Improved performance of simulations using chemeq with dynamic rebalance. Specifically, the default single weight criteria per equation system used by the graph partitioner was split into several weight criteria (`CHEMEQ`, `ASSEMBLY`, `LINEAR_SOLVE`, `DEFAULT`) resulting in better load balancing.
- Increased the maximum number of variables allowed in a string function to 50 variables.
- Stability improvements for Sparc-Aria aerothermal coupling.

- Added native implementation of (micromechanics pressurization) MMP modeling within Aria. This model incorporates decomposition/thermal expansion of pores and tracks material permeability within explosive materials using native expressions. This model is based off of the work done by Hobbs et al. [2].

Convention Change

- Updated sign convention for residual-based post-processors to be consistent with expression-based post-processors and aria flux specifications (positive = outflux)

2.14 Version 5.4 (2022/02/14)

New Aria Features

- Improved conservation errors for complex advective bar geometries. A new advection velocity model has been added `advection velocity = bar`. This model computes the velocity field directly at each integration point while ensuring proper alignment with the bar flow direction. The previous user field model interpolated a nodal velocity field to the integration points, which could result in large conservation errors for non-collinear bars. This previous model `advection velocity = user_field name = bar_velocity` has been deprecated and will error in the 5.8 release.

New Chaparral Features

- Added memory estimates and suggestions for enclosures to the chaparral log file. Also added a command line keyword to terminate the simulation early after outputting these estimates/suggestions to the chaparral log file. This can be done by adding `--estimate_viewfactor_proc` after the aria executable.

Bug Fixes

- Fixed an issue with heartbeat output on simulation end. Aria will now always output the final step of a simulation e.g. if `Termination Time = 5` and the heartbeat interval is `at time 0 interval is 2`, time=5 will still appear in all output. This is now true of all output (i.e. the heartbeat, exodus output, restart database, etc)
- Fixed an issue in the advective bar where the velocity was computed using a density based on the temperature at the previous time step as opposed to the iterated temperature field available from the nonlinear iteration.

New Deprecations (will error in 5.4)

- The `getGlobalVariable(name)` interface for user subroutines is deprecated use `getGlobalVariableValue(name)` instead. The old method returned the current global variable reference, which allowed unsafe editing and potential bugs.
- The use of `USE BLOCK` within the bulk element block to specify a block in the mesh to be used in lieu of creating bulk node parts has been deprecated. If a mesh already

contains a bulk node block with the same name (e.g. from a previous simulation) this will be used automatically.

- Output to file support for dataprobes has been deprecated. Please add the data probe global variable to a heartbeat output block instead.

2.15 Version 5.2 (2021/11/01)

New Features

- Added capability to postprocess global variables using global string functions

  These global functions execute in the order they are parsed, so take care when using global functions that depend on other global functions. Global functions are evaluated last among all of the other postprocess commands

- Added capability to define an arbitrary binary state user field, with optional activation and deactivation criteria specified as string functions. Within a material block, these fields can be defined using

Bug Fixes

- Refactored global variable interface to fix internal bugs and remove error-prone interfaces. Note that the `getGlobalVariable(name)` interface for user subroutines is deprecated use `getGlobalVariableValue(name)` instead. The old method returned the current global variable reference, which allowed unsafe editing and potential bugs.

- Bug fixed regarding toggling multiple contact enforcements independently (COMPSIMHD-10544). Parsing now enforces a one-to-one relationship between the contact definition block and the enclosed enforcement block.

  #### Known issues

  - Contact status and face visualization fields are potentially incorrect when a surface is shared between separate contact definition blocks that are toggle independently.

- Bug fixed regarding element activation when using directed energy model (COMPSIMHD-10518). The model now activates all elements whose average temperature exceeds the activation temperature if given. Note that some slight differences in solution are possible, as the previous approach activated elements when any single node exceeded the activation temperature. To avoid activating elements prematurely before they are deposited, both the directed energy source and activation_user_function thermal conductivities now support referring to a 'deposit' field.

Upcoming Deprecations (will warn in 5.2)

- The `getGlobalVariable(name)` interface for user subroutines is deprecated use `getGlobalVariableValue(name)` instead. The old method returned the current global variable reference, which allowed unsafe editing and potential bugs.
- The use of `USE BLOCK` within the bulk element block to specify a block in the mesh to be used in lieu of creating bulk node parts has been deprecated. If a mesh already contains a bulk node block with the same name (e.g. from a previous simulation) this will be used automatically.
- Output to file support for dataprobes has been deprecated. Please add the data probe global variable to a heartbeat output block instead.

2.16 Version 5.0 (2021/04/26)

Linear Solvers

- Added capability to dynamically compute linear residual convergence tolerances during execution. The goal of this capability is to reduce the number of linear iterations required to achieve nonlinear convergence, therefore decreasing computational costs.
- Added beta capability to use block preconditioning through the Trilinos library Teko.

Participating Media Radiation

- Added support for MPMD coupling with Nalu to model participating media radiation.

Postprocessing

- Added new syntax for various postprocessing operations that previously required using more cumbersome Encore syntax.

Advective Bar

- Enable pressure solve for the bar network including kinetic energy, wall friction loss, K factors and hydrostatic contributions.

Bug fixes

- Fixed a bug where using a lumped flux BC on the surface of a tet mesh would apply 6x the intended flux.

A timeline of upcoming deprecations is provided in the figure below

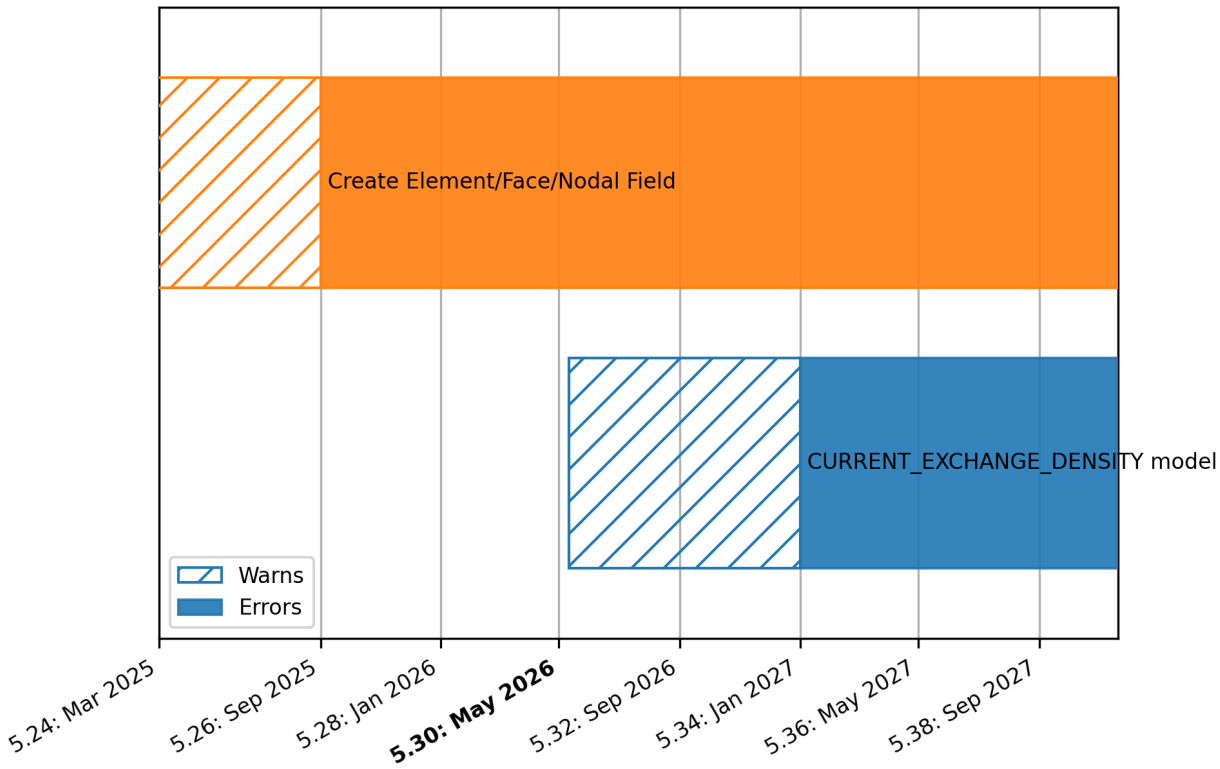


Fig. 2.1: Aria Deprecation Timeline

3 Theory

3.1 Domain Definition

To fix the notation, consider Fig. 3.1, which is a schematic representation of a typical problem. The entire domain is represented by Ω, which, for example, lies in three-dimensional coordinate space, with spatial coordinates $\mathbf{x}$. In this particular case, Ω consists of two separate subdomains, $\Omega = \Omega_1 \cup \Omega_2$. These subdomains may consist of different materials. The entire boundary of Ω is indicated by $\partial\Omega$, subject to one or more boundary conditions on subsections we denote with a subscript on Γ. For example, let Γ_q be that portion of $\partial\Omega$ along which a specified heat flux normal to the boundary is applied; similarly, let Γ_T be subject to an applied temperature; let the surface Γ_a be adiabatic (no heat flux); let Γ_r, be subject to an applied radiation heat flux; and let Γ_h be subject to a convective heat flux, which is modeled by Newton's law of cooling. Note that the boundary conditions are of two types: either the flux or the temperature is specified. Finally, the interface between Ω_1 and Ω_2 is denoted $\partial\Omega_{1-2}$. The interface conditions applied along a boundary such as $\partial\Omega_{1-2}$ are that both the temperature and the normal component of the heat flux are continuous. Given appropriate initial conditions, the problem is to determine the time-evolution of the temperature field.

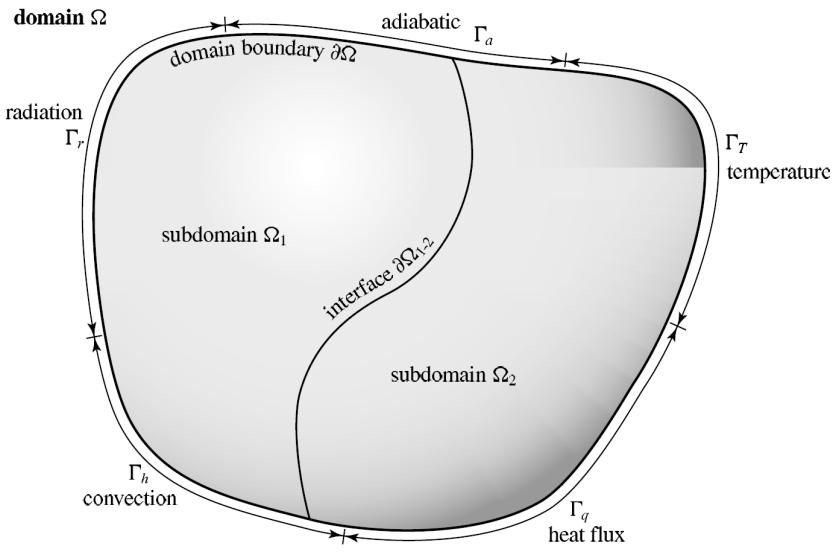


Fig. 3.1: A schematic diagram of the mathematical thermal model, showing the domain Ω; the subdomains Ω_i and their interfaces $\partial\Omega_{i-j}$; and the boundary conditions on the surface Γ.

Note that when referencing a generic volume or surface, V and S will be used respectively.

3.2 Governing Equations

Generalized Conservation Equation

We first introduce a general conservation equation, as a model for the specific equations that Aria solves, demonstrating how the Galerkin finite element method is applied to it, and how the integration by parts is carried out on its individual terms. Following [3], the conservation of a general scalar quantity $\phi(\boldsymbol{x}, t)$, with units of amount-per-unit-volume, at a point $\boldsymbol{x}$ and time t can be expressed as

$$\frac{\partial \phi}{\partial t} + \boldsymbol{\nabla} \cdot (\phi \boldsymbol{v}) = -\boldsymbol{\nabla} \cdot \boldsymbol{f} + \phi_{\mathrm{V}} \tag{3.1}$$

where $\boldsymbol{v}$ is the mass average velocity, $\boldsymbol{f}$ is the diffusive flux of ϕ, and ϕ_{V} is the volumetric source of ϕ. In *Equations Aria Solves*, the physical meaning of the source term of each equation is discussed.

The Galerkin FEM (G/FEM) residual form of (3.1) is formed by bringing the right hand side terms to the left, multiplying by the FEM weight function ψ^i and integrating over the volume V,

$$R^i_\phi = \int_{\mathrm{V}} \left(\frac{\partial \phi}{\partial t} + \boldsymbol{v} \cdot \boldsymbol{\nabla} \phi + \phi \boldsymbol{\nabla} \cdot \boldsymbol{v} + \boldsymbol{\nabla} \cdot \boldsymbol{f} - B_{\mathrm{V}} \right) \psi^i \, \mathrm{dV} = 0. \tag{3.2}$$

In many applications $\boldsymbol{\nabla} \cdot \boldsymbol{v} = 0$ so we ignore that term from here on. However, it is straight forward to account for this term via the source term B_{V}. Using the vector identity $(\boldsymbol{\nabla} \cdot \boldsymbol{f})\psi^i = \boldsymbol{\nabla} \cdot (\boldsymbol{f}\psi^i) - \boldsymbol{\nabla}\psi^i \cdot \boldsymbol{f}$ and using the divergence theorem, (3.2) becomes

$$R^i_\phi = \int_{\mathrm{V}} \left[\left(\frac{\partial \phi}{\partial t} + \boldsymbol{v} \cdot \boldsymbol{\nabla} \phi - B_{\mathrm{V}} \right) \psi^i - \boldsymbol{\nabla}\psi^i \cdot \boldsymbol{f} \right] \mathrm{dV} + \int_{\mathrm{S}} \boldsymbol{n} \cdot \boldsymbol{f} \psi^i \, \mathrm{dS} = 0. \tag{3.3}$$

Here $\boldsymbol{n}$ is a unit normal along the boundary S, pointing out of the volume V.

Equation (3.3) embodies the sign convention for sources, fluxes and equation terms used within Aria. For example, scalar flux expressions in Aria provide values for $f_n \equiv \boldsymbol{n} \cdot \boldsymbol{f}$ and should be positive for a flux of ϕ leaving the volume V.

Note also that we have not assigned a units convention to the equation. Any unit system may be employed in the specification of the individual terms in (3.1). However, each term in (3.1) must have overall units of [ϕ] / [time], and the overall residual expression has units of [ϕ] * [L]^3/ [time], where [ϕ] are units of the conserved quantity, ϕ, [L] is the unit of the length scale, and [time] is the unit for time.

Statement of the Transient Problem

We are now in a position to state mathematically the initial–boundary value problem described at the beginning of this chapter. Let the domain Ω consist of N non-overlapping subdomains. Let $\mathcal{T}$ be the set of all subdomains. Then the statement of the boundary value problem becomes, find the solution $\phi = \phi(\mathbf{x}, t)$, which satisfies

$$\begin{aligned}
\frac{\partial \phi}{\partial t} + \boldsymbol{v} \cdot \boldsymbol{\nabla}\phi - \boldsymbol{\nabla} \cdot \boldsymbol{f} &= \phi_V \quad \forall\, \mathbf{x} \in \{\Omega_i \,|\, \Omega_i \in \mathcal{T}\}; t > t_0 \\
\phi(\mathbf{x}, t_0) &= \phi_0(\mathbf{x}) \qquad \forall\, \mathbf{x} \in \Omega \\
\phi(\mathbf{x}, t) &= \phi_b(\mathbf{x}, t) \qquad \forall\, \mathbf{x} \in \Gamma_\phi \\
f_n &= f_b(\mathbf{x}, t, \phi) \qquad \forall\, \mathbf{x} \in \partial\Omega \setminus \Gamma_\phi \\
[\![f_n]\!] &= 0 \quad \forall\, \mathbf{x} \in \{\partial\Omega_{i-k} \,|\, \Omega_i, \Omega_k \in \mathcal{T}\}
\end{aligned} \tag{3.4}$$

Here f_n is the component of flux normal to a surface, where $f_n = \boldsymbol{f} \cdot \boldsymbol{n}$, and n is the outward unit normal vector. The notation $\partial\Omega \setminus \Gamma_\phi$ indicates the complement of Γ_ϕ in $\partial\Omega$,

$$\partial\Omega \setminus \Gamma_\phi = \left\{ \mathbf{x} \in \partial\Omega \mid \mathbf{x} \notin \Gamma_\phi \right\},$$

or the boundary of Ω excluding the surface Γ_ϕ. In order for the problem to be well posed, it is not possible to specify both the flux and the temperature at the same location. Note that the initial condition ϕ_0, the boundary condition ϕ, and the flux boundary condition f_b are usually specified in a piecewise manner over the subdomains and their boundaries. Various forms of the specified flux function $f_b(\mathbf{x}, t, \phi)$ are possible and will be described later in this chapter. For example, an adiabatic condition is specified if $f_b(\mathbf{x}, t, \phi) = 0$. The notation $[\![f_n]\!]$ represents the jump in the normal flux across a surface between two subdomains.

Statement of the Stationary Problem

For stationary or steady-state problems (3.4) may be simplified since the time derivative vanishes by definition. Accordingly, we obtain

$$\begin{aligned}
\boldsymbol{v} \cdot \boldsymbol{\nabla}\phi - \boldsymbol{\nabla} \cdot \boldsymbol{f} &= \phi_V \quad \forall\, \mathbf{x} \in \{\Omega_i \,|\, \Omega_i \in \mathcal{T}\}; t > t_0 \\
\phi(\mathbf{x}) &= \phi_b(\mathbf{x}) \qquad \forall\, \mathbf{x} \in \Gamma_\phi \\
f_n &= f_b(\mathbf{x}, \phi) \qquad \forall\, \mathbf{x} \in \partial\Omega \setminus \Gamma_\phi \\
[\![f_n]\!] &= 0 \quad \forall\, \mathbf{x} \in \{\partial\Omega_{i-k} \,|\, \Omega_i, \Omega_k \in \mathcal{T}\}
\end{aligned} \tag{3.5}$$

Equations Aria Solves

Below, a more detailed description is provided for each equation currently supported by Aria:

Conservation of Mass

For a material with density ρ, letting $b = \rho$ results in the conservation of mass. Since there is no net flow relative to the mass average velocity $\boldsymbol{f} = \mathbf{0}$. Although there are no sources of mass, having such a source can be convenient in modeling and simulation; so, we let the mass source be $B_{\mathrm{V}} = q_m$. Thus, equation (3.1) becomes

$$\frac{\partial \rho}{\partial t} + \rho \boldsymbol{\nabla} \cdot \boldsymbol{v} + \boldsymbol{v} \cdot \boldsymbol{\nabla} \rho = q_m. \tag{3.6}$$

For the special but common case of constant density, this reduces to

$$\boldsymbol{\nabla} \cdot \boldsymbol{v} = 0. \tag{3.7}$$

Using equation (3.3), the G/FEM residual form is

$$R_P^i = \int_{\mathrm{V}} \left(-\frac{\partial \rho}{\partial t} - \rho \boldsymbol{\nabla} \cdot \boldsymbol{v} - \boldsymbol{v} \cdot \boldsymbol{\nabla} \rho + q_m \right) \phi^i \, \mathrm{dV} = 0. \tag{3.8}$$

Attention: Equation (3.8) has been multiplied by -1 because this form results in a better linear system for the special case of incompressible flow. This is important to remember when defining mass source terms.

In Aria, each term in (3.8) is specified separately as identified in equation (3.9).

$$R_P^i = \underbrace{\int_{\mathrm{V}} -\frac{\partial \rho}{\partial t} \phi^i \, \mathrm{dV}}_{\text{MASS}} + \underbrace{\int_{\mathrm{V}} -\left(\boldsymbol{v} \cdot \boldsymbol{\nabla} \rho + \rho \boldsymbol{\nabla} \cdot \boldsymbol{v} \right) \phi^i \, \mathrm{dV}}_{\text{ADV}} + \underbrace{\int_{\mathrm{V}} q_m \phi^i \, \mathrm{dV}}_{\text{SRC}} = 0 \tag{3.9}$$

For a purely incompressible form, Aria offers the alternative form given in (3.10);

$$R_P^i = \underbrace{\int_{\mathrm{V}} -\boldsymbol{\nabla} \cdot \boldsymbol{v} \phi^i \, \mathrm{dV}}_{\text{DIV}} + \underbrace{\int_{\mathrm{V}} q_m \phi^i \, \mathrm{dV}}_{\text{SRC}} = 0 \tag{3.10}$$

Conservation of Energy

For a material with constant density and specific heat C_p, temperature T, heat flux $\boldsymbol{q}$ and volumetric energy source H_V, letting $b = \rho C_p T$, $\boldsymbol{f} = \boldsymbol{q}$ and $B_\mathrm{V} = H_V$ results in the conservation of energy.

$$\rho C_p \frac{\partial T}{\partial t} + \rho C_p \boldsymbol{v} \cdot \boldsymbol{\nabla} T = -\boldsymbol{\nabla} \cdot \boldsymbol{q} + H_V. \tag{3.11}$$

A common constitutive relationship for $\boldsymbol{q}$ is Fourier's law, $\boldsymbol{q} = -\kappa \boldsymbol{\nabla} T$ where κ is the thermal conductivity. However, we leave the heat flux as an option to be specified as part of the material properties (see *Heat Conduction*). Using equation (3.3), the G/FEM residual form is

$$R_T^i = \int_\mathrm{V} \left[\left(\rho C_p \frac{\partial T}{\partial t} + \rho C_p \boldsymbol{v} \cdot \boldsymbol{\nabla} T - H_V \right) \phi^i - \boldsymbol{\nabla} \phi^i \cdot \boldsymbol{q} \right] \mathrm{dV} + \int_\mathrm{S} q_n \phi^i \, \mathrm{dS} = 0 \tag{3.12}$$

where q_n is the heat flux at the boundary. For example, the natural convection boundary condition gives $q_n = h(T - T_\infty)$ where h is the heat transfer coefficient and T_∞ is the bulk temperature away from the surface.

In Aria, each term in (3.12) is specified separately as identified in equation (3.13).

$$\begin{aligned} R_T^i = & \underbrace{\int_\mathrm{V} \rho C_p \frac{\partial T}{\partial t} \phi^i \, \mathrm{dV}}_{\mathrm{MASS}} + \underbrace{\int_\mathrm{V} \rho C_p \boldsymbol{v} \cdot \boldsymbol{\nabla} T \phi^i \, \mathrm{dV}}_{\mathrm{ADV}} + \underbrace{\int_\mathrm{V} \rho C_p T \boldsymbol{\nabla} \cdot \boldsymbol{v} \phi^i \, \mathrm{dV}}_{\mathrm{DIV}} \\ & - \underbrace{\int_\mathrm{V} H_V \phi^i \, \mathrm{dV}}_{\mathrm{SRC}} - \underbrace{\int_\mathrm{V} \boldsymbol{\nabla} \phi^i \cdot \boldsymbol{q} \, \mathrm{dV}}_{\mathrm{DIFF}} + \int_\mathrm{S} q_n \phi^i \, \mathrm{dS} = 0 \end{aligned} \tag{3.13}$$

For a purely incompressible form, we know that $\boldsymbol{\nabla} \cdot \boldsymbol{v} \equiv 0$ and can remove terms. Equation (3.13) then simplifies to

$$\begin{aligned} R_T^i = & \underbrace{\int_\mathrm{V} \rho C_p \frac{\partial T}{\partial t} \phi^i \, \mathrm{dV}}_{\mathrm{MASS}} + \underbrace{\int_\mathrm{V} \rho C_p \boldsymbol{v} \cdot \boldsymbol{\nabla} T \phi^i \, \mathrm{dV}}_{\mathrm{ADV}} - \underbrace{\int_\mathrm{V} H_V \phi^i \, \mathrm{dV}}_{\mathrm{SRC}} \\ & - \underbrace{\int_\mathrm{V} \boldsymbol{\nabla} \phi^i \cdot \boldsymbol{q} \, \mathrm{dV}}_{\mathrm{DIFF}} + \int_\mathrm{S} q_n \phi^i \, \mathrm{dS} = 0 \end{aligned} \tag{3.14}$$

Note, however, that both equations still assume a constant specific heat C_p.

SUPG

SUPG stabilization can be activated by added the `SUPG` term on the input file line. This capability is still under development.

Conservation of Chemical Species

For a material with species k with molar concentration C_k, molar flux $\boldsymbol{J}_k$ relative to the mass average velocity and volumetric reaction rate $R_{V,k}$, letting $b = y_k$, $\boldsymbol{f} = \boldsymbol{J}_k$ and $B_\mathrm{V} = R_{V,k}$ in equation (3.1) results in the conservation equation for species k,

$$\frac{\partial C_k}{\partial t} + \boldsymbol{v} \cdot \boldsymbol{\nabla} C_k = -\boldsymbol{\nabla} \cdot \boldsymbol{J}_k + R_{V,k}. \tag{3.15}$$

For liquid mixtures which are dilute in all species except one, Fick's law is often used to approximate $\boldsymbol{J}_k$. In this approximation, D_k represents the diffusion coefficient of species k with respect to the concentrated species and it is assumed that the interactions between dilute species is assumed negligible. Again, however, we choose to leave the governing equation in the more general form and require the particular diffusive flux model as user input (see *Species Diffusion*). Using equation (3.3), the G/FEM residual form is

$$R^i_{C_k} = \int_\mathrm{V} \left[\left(\frac{\partial C_k}{\partial t} + \boldsymbol{v} \cdot \boldsymbol{\nabla} C_k - R_{V,k} \right) \phi^i - \boldsymbol{\nabla} \phi^i \cdot \boldsymbol{J}_k \right] \mathrm{dV} + \int_\mathrm{S} q_{n,k} \phi^i \, \mathrm{dS} = 0 \tag{3.16}$$

where $q_{n,k}$ is the mass flux at the boundary. For example, the natural convection boundary condition gives $q_n = k(C_k - C_{\infty,k})$ where k is the mass transfer coefficient and $C_{\infty,k}$ is the bulk concentration away from the surface.

In Aria, each term in (3.16) is specified separately as identified in equation (3.17).

$$\begin{aligned} R^i_{C_k} = & \underbrace{\int_\mathrm{V} \frac{\partial C_k}{\partial t} \phi^i \, \mathrm{dV}}_{\text{MASS}} + \underbrace{\int_\mathrm{V} \boldsymbol{v} \cdot \boldsymbol{\nabla} C_k \phi^i \, \mathrm{dV}}_{\text{ADV}} + \underbrace{\int_\mathrm{V} C \boldsymbol{\nabla} \cdot \boldsymbol{v} \phi^i \, \mathrm{dV}}_{\text{DIV}} \\ & \underbrace{- \int_\mathrm{V} R_{V,k} \phi^i \, \mathrm{dV}}_{\text{SRC}} \underbrace{- \int_\mathrm{V} \boldsymbol{\nabla} \phi^i \cdot \boldsymbol{J}_k \, \mathrm{dV}}_{\text{DIFF}} + \int_\mathrm{S} q_{n,k} \phi^i \, \mathrm{dS} = 0 \end{aligned} \tag{3.17}$$

For a purely incompressible form, we know that $\boldsymbol{\nabla} \cdot \boldsymbol{v} \equiv 0$ and can remove terms. Equation (3.17)

then simplifies to

$$R^i_{C_k} = \underbrace{\int_V \frac{\partial C_k}{\partial t}\phi^i \, \mathrm{dV}}_{\text{MASS}} + \underbrace{\int_V \boldsymbol{v}\cdot\boldsymbol{\nabla} C_k \phi^i \, \mathrm{dV}}_{\text{ADV}} - \underbrace{\int_V R_{V,k}\phi^i \, \mathrm{dV}}_{\text{SRC}} - \underbrace{\int_V \boldsymbol{\nabla}\phi^i \cdot \boldsymbol{J}_k \, \mathrm{dV}}_{\text{DIFF}} + \int_S q_{n,k}\phi^i \, \mathrm{dS} = 0 \tag{3.18}$$

Often times it is useful to solve for mass, weight or volume fractions of each species rather than for the concentration directly. In that case, an additional condition exists,

$$\sum_k C_k = 1 \tag{3.19}$$

Using this condition, it is only necessary to solve for $N-1$ species fractions where N is the total number of species present in the problem. The final species, then, is simply given as

$$C_j = 1 - \sum_{k\neq j} C_k \tag{3.20}$$

This method can be triggered in Aria by specifying the equation term `FRACBAL`. In this case, the equation for C_j is not included in the system of unknowns but is instead post-processed on the fly. Aria will automatically detect all other species equations and include them in the fraction balance.

Conservation of Fluid Momentum

The Cauchy momentum equation is given by

$$\rho\frac{\partial \boldsymbol{v}}{\partial t} + \rho\boldsymbol{v}\cdot\boldsymbol{\nabla}\boldsymbol{v} - \boldsymbol{g} - \boldsymbol{\nabla}\cdot\boldsymbol{T} = \boldsymbol{0} \tag{3.21}$$

where $\boldsymbol{T}$ is the fluid stress tensor and $\boldsymbol{g}$ is a body force. We construct the G/FEM residual form of (3.21) by contracting with the unit coordinate vector in the k-direction, $\boldsymbol{e}_k$, multiplying by the weight function ϕ^i and integrating over the volume. Using the vector identity $(\boldsymbol{\nabla}\cdot\boldsymbol{T})\cdot\boldsymbol{e}_k\phi^i = \boldsymbol{\nabla}\cdot(\boldsymbol{T}\cdot\boldsymbol{e}_k\phi^i) - \boldsymbol{T}^t : \boldsymbol{\nabla}(\boldsymbol{e}_k\phi^i)$ and integrating by parts gives

$$R^i_{m,k} = \int_V \left[\left(\rho\frac{\partial \boldsymbol{v}}{\partial t} + \rho\boldsymbol{v}\cdot\boldsymbol{\nabla}\boldsymbol{v} - \boldsymbol{g}\right)\cdot\boldsymbol{e}_k\phi^i + \boldsymbol{T}^t : \boldsymbol{\nabla}\left(\boldsymbol{e}_k\phi^i\right)\right] \mathrm{dV} - \int_S \boldsymbol{n}\cdot\boldsymbol{T}\cdot\boldsymbol{e}_k\phi^i \, \mathrm{dS} = 0 \tag{3.22}$$

In Aria, each term in (3.22) is specified separately as identified in equation (3.23).

$$\begin{aligned} R^i_{m,k} = & \underbrace{\int_V \rho \frac{\partial \boldsymbol{v}}{\partial t} \cdot \boldsymbol{e}_k \phi^i \, \mathrm{dV}}_{\text{MASS}} + \underbrace{\int_V \rho \boldsymbol{v} \cdot \boldsymbol{\nabla} \boldsymbol{v} \cdot \boldsymbol{e}_k \phi^i \, \mathrm{dV}}_{\text{ADV}} - \underbrace{\int_V \boldsymbol{g} \cdot \boldsymbol{e}_k \phi^i \, \mathrm{dV}}_{\text{SRC}} \\ & + \underbrace{\int_V \boldsymbol{T}^t : \boldsymbol{\nabla} \left(\boldsymbol{e}_k \phi^i \right) \, \mathrm{dV}}_{\text{DIFF}} - \int_S \boldsymbol{n} \cdot \boldsymbol{T} \cdot \boldsymbol{e}_k \phi^i \, \mathrm{dS} = 0 \end{aligned} \tag{3.23}$$

Conservation of Solid Momentum

Aria currently solves the quasistatic form of the solid momentum equations. Furthermore, the solid stress is treated as a linear elastic material. In this limit, the Cauchy momentum equation is given by

$$\boldsymbol{\nabla} \cdot \boldsymbol{T} = \boldsymbol{0} \tag{3.24}$$

where $\boldsymbol{T}$ is the solid stress tensor. We construct the G/FEM residual form of (3.21) by contracting with the unit coordinate vector in the k-direction, $\boldsymbol{e}_k$, multiplying by the weight function ϕ^i and integrating over the volume. Using the vector identity
$(\boldsymbol{\nabla} \cdot \boldsymbol{T}) \cdot \boldsymbol{e}_k \phi^i = \boldsymbol{\nabla} \cdot (\boldsymbol{T} \cdot \boldsymbol{e}_k \phi^i) - \boldsymbol{T}^t : \boldsymbol{\nabla}(\boldsymbol{e}_k \phi^i)$ and integrating by parts gives

$$R^i_{m,k} = \int_V \boldsymbol{T}^t : \boldsymbol{\nabla} \left(\boldsymbol{e}_k \phi^i \right) \, \mathrm{dV} = 0 \tag{3.25}$$

Here, the surface contribution, $\int_S \boldsymbol{n} \cdot \boldsymbol{T} \cdot \boldsymbol{e}_k \phi^i \, \mathrm{dS}$, has been dropped because Aria currently only supports Dirichlet and natural (homogeneous) boundary conditions for the solid equation.

In Aria, each term in (3.25) is specified separately as identified in equation (3.26).

$$R^i_{m,k} = \underbrace{\int_V \boldsymbol{T}^t : \boldsymbol{\nabla} \left(\boldsymbol{e}_k \phi^i \right) \, \mathrm{dV}}_{\text{DIFF}} = 0 \tag{3.26}$$

Currently, Aria does not support direct specification of the more popular stress-strain parameterization that utilizes Young's modulus E, Poisson's ratio ν and coefficient of thermal expansion α (note, the shear modulus $G = \mu$). The relationship between these two parameterizations is summarized here for convenience.

$$\begin{aligned} 2\mu &= \frac{E}{(1+\nu)} \\ \lambda &= \frac{\nu E}{(1+\nu)\,(1-2\nu)} &&= 2\mu \frac{\nu}{(1-2\nu)} \\ \beta &= \frac{\alpha E}{(1-2\nu)} &&= \alpha\,(3\lambda + 2\mu) \end{aligned}$$

When the solid momentum equation is solved, the physical coordinates of the region, used to calculate integration quantities, is deformed by the displacement vector. The current configuration, $\mathbf{x}$, is calculated from the reference configuration, $\mathbf{X}$, by

$$\mathbf{x} = \mathbf{X} + \mathbf{d},$$

where $\mathbf{d}$ is the displacement field. The displacement field is chosen automatically by the code, but may be overridden by setting a solution option. These displaced, physical coordinates are used for integration of all of the equations.

Additionally, the code supports the integration of the solid momentum equations in the reference configuration, rather than in the current configuration. This is activated through the use of the `REF_DIFF` equation term rather than the `DIFF` term. When this is used, the Cauchy stress tensor is rotated into the Second Piola-Kirchhoff stress tensor. Integration quantities are also calculated in the reference configuration. Care should be taken to ensure that the constitutive model used also changes its strain measure to one which is work conjugate with the Second Piola-Kirchhoff stress tensor.

If both the `MESH` and `SOLID` equations are defined in the same region, then the TALE mesh motion algorithm is activated automatically. Here, the reference frame for the `SOLID` equation is changed to the TALE reference frame, which is defined as

$$\underline{\mathbf{x}}_r = \underline{\mathbf{X}} + \underline{\mathbf{d}}_m - \underline{\mathbf{d}}_s$$

where $\underline{\mathbf{x}}_r$ is the new TALE reference frame, $\underline{\mathbf{d}}_m$ is `MESH_DISPLACEMENTS`, and $\underline{\mathbf{d}}_s$ is `SOLID_DISPLACEMENTS`. This affects the calculation of strain measures, such as $\underline{\underline{F}}$, and integration quantities, such as $\det(J)$, among others. More details of the TALE method will be available soon in a pending report.

Radiation Transport Equation

The transport of radiative intensity in a given direction $I(s)$ through an absorbing medium is governed by the radiation transport equation.

$$s \cdot \nabla I(s) + \mu_a I(s) = S \tag{3.27}$$

Where s_i is an ordinate direction, μ_a is the absorption cross section, and S is the radiation source. Note that Aria currently does not support scattering terms for the radiation transport equation, so they have not been included here.

The radiation transport equation is not solved in Aria, but rather through an MPMD coupling to a PMR application. An MPMD coupling block in the Aria region is required to activate this equation (see *MPMD Coupling*). The only valid term for this equation is `XFER`, and this equation specification is simply used to activate the MPMD coupling for the selected mesh part.

Voltage Equation

The electric potential or voltage V is frequently used in determining the electric field, $\boldsymbol{E} = -\boldsymbol{\nabla} V$. While equation (3.1) cannot be applied to the voltage, the equation governing the voltage – Gauss' law from Maxwell's equations – has a similar form. Writing the electric displacement $\boldsymbol{D}$ as $\boldsymbol{D} = \epsilon \boldsymbol{E}$, where ϵ is the electric permittivity, Gauss' law is

$$-\boldsymbol{\nabla} \cdot \epsilon \boldsymbol{\nabla} V = \rho_e \tag{3.28}$$

where the permittivity is taken to be a constant and ρ_e is the volumetric free charge density.

Using equation (3.3), the G/FEM residual form is

$$R_V^i = \int_{\mathrm{V}} \left(-\rho_e \phi^i + \boldsymbol{\nabla} \phi^i \cdot \epsilon \boldsymbol{\nabla} V \right) \, \mathrm{dV} + \int_{\mathrm{S}} q_n \phi^i \, \mathrm{dS} = 0 \tag{3.29}$$

In Aria, each term in (3.29) is specified separately as identified in equation (3.30).

$$R_V^i = \underbrace{-\int_{\mathrm{V}} \rho_e \phi^i \, \mathrm{dV}}_{\mathrm{SRC}} + \underbrace{\int_{\mathrm{V}} \boldsymbol{\nabla} \phi^i \cdot \epsilon \boldsymbol{\nabla} V \, \mathrm{dV}}_{\mathrm{DIFF}} + \int_{\mathrm{S}} q_n \phi^i \, \mathrm{dS} = 0 \tag{3.30}$$

Current Equation

An alternate formulation in solving for the electrical potential (see *Voltage Equation*) is to solve the "current" equation which is a conservation equation for electrical charge. The electrical current $\boldsymbol{J}$ is frequently related to the electric field $\boldsymbol{E}$ using Ohm's law as $\boldsymbol{J} = \sigma_e \boldsymbol{E}$ where σ_e is the electrical conductivity. The electric potential or voltage V is used in determining the electric field, $\boldsymbol{E} = -\boldsymbol{\nabla} V$. However, we choose to leave the electrical current as a more general constitutive model to be provided as a material model input (see *Current Density*). More complex version for charged ion fluxes (Nerst Planck, etc. can be utilized with the species equation).

$$-\boldsymbol{\nabla} \cdot \boldsymbol{J} = S \tag{3.31}$$

Here S is a current source term, for example from electrochemical reactions in a battery simulation.

Using equation (3.3), the G/FEM residual form is

$$R_V^i = \int_{\mathrm{V}} \left(-S \phi^i - \boldsymbol{\nabla} \phi^i \cdot \boldsymbol{J} \right) \, \mathrm{dV} + \int_{\mathrm{S}} q_n \phi^i \, \mathrm{dS} = 0 \tag{3.32}$$

In Aria, each term in (3.32) is specified separately as identified in equation (3.33).

$$R_V^i = \underbrace{-\int_{\mathrm{V}} S \phi^i \, \mathrm{dV}}_{\mathrm{SRC}} \underbrace{- \int_{\mathrm{V}} \boldsymbol{\nabla} \phi^i \cdot \boldsymbol{J} \, \mathrm{dV}}_{\mathrm{DIFF}} + \int_{\mathrm{S}} q_n \phi^i \, \mathrm{dS} = 0 \tag{3.33}$$

Charge Density Equation

This equation solves for the volumetric free charge density, ρ_e, and is meant to be coupled with the voltage equation in *Voltage Equation*. This equation can also be used in place of the current equation (3.31), since this equation has the correct form (the current equation has the correct form with ρ_e when electric displacement is activated) and the charge density is treated as a variable. The charge density evolves as

$$\frac{\partial \rho_e}{\partial t} + \boldsymbol{v} \cdot \boldsymbol{\nabla} \rho_e = -\boldsymbol{\nabla} \cdot \boldsymbol{J} = \boldsymbol{\nabla} \cdot (\sigma \boldsymbol{\nabla} V) \tag{3.34}$$

where $\boldsymbol{J} = -\sigma \boldsymbol{\nabla} V$ is assumed to obey Ohm's law and σ represents the conductivity.

The G/FEM residual form of this equation is

$$R_V^i = \int_{\mathrm{V}} \left(\frac{\partial \rho_e}{\partial t} \phi^i + \boldsymbol{\nabla} \phi^i \cdot \boldsymbol{v} + \boldsymbol{\nabla} \phi^i \cdot \sigma \boldsymbol{\nabla} V \right) \mathrm{dV} + \int_{\mathrm{S}} q_n \phi^i \, \mathrm{dS} = 0 \tag{3.35}$$

In Aria, each term in (3.35) is specified separately as identified in equation (3.36).

$$R_V^i = \underbrace{\int_{\mathrm{V}} \frac{\partial \rho_e}{\partial t} \phi^i \, \mathrm{dV}}_{\text{MASS}} + \underbrace{\int_{\mathrm{V}} \boldsymbol{\nabla} \phi^i \cdot \boldsymbol{v} \, \mathrm{dV}}_{\text{ADV}} + \underbrace{\int_{\mathrm{V}} \boldsymbol{\nabla} \phi^i \cdot \sigma \boldsymbol{\nabla} V \, \mathrm{dV}}_{\text{DIFF}} + \int_{\mathrm{S}} q_n \phi^i \, \mathrm{dS} = 0. \tag{3.36}$$

Ampere Equation

Ampere's equation in a conductive material is given by

$$\nabla \times \boldsymbol{H} = \boldsymbol{J} + \frac{\partial \boldsymbol{D}}{\partial t}$$

where $\boldsymbol{H}$ is the magnetic field intensity, $\boldsymbol{J}$ is the current density and $\boldsymbol{D}$ is the electric displacement. Ignoring time dependent change in electric displacement a common approach of solving Ampere's equation for a magnetic flux intensity $\mathbf{H}$ in a conductive material is to recast the formulation in terms of magnetic field flux $\boldsymbol{B}$ by employing the constitutive equation relating magnetic field intensity to magnetic field flux, permeability of free space μ_o and magnetization vector $\boldsymbol{M}$, $\mathbf{H} = \frac{1}{\mu_o} \boldsymbol{B} - \boldsymbol{M}$ subject to the constraint $\nabla \cdot \boldsymbol{B}$. An alternative formulation of this problem is obtained by introducing a magnetic vector potential $\boldsymbol{A}$ where $\nabla \times \boldsymbol{A} = \boldsymbol{B}$. This approach circumvents the need to explicitly impose the divergence free constraint $\nabla \cdot \boldsymbol{B}$ required in the formulation of the magnetic flux density. For static configurations the current density is a function of the electric field $\boldsymbol{E}$ and the electrical conductivity σ, $\boldsymbol{J} = \sigma \cdot \boldsymbol{E}$ whereas for conductors moving with velocity $\boldsymbol{v}$ $\boldsymbol{J} = \sigma \cdot \boldsymbol{E} + \sigma \cdot (\boldsymbol{v} \times \boldsymbol{B})$. Hence the formulation for moving conductors will include contributions obtained from the solution of Ohm's law for voltage V (see *Voltage Equation*).

$$\frac{\partial \boldsymbol{A}}{\partial t} - \boldsymbol{v} \times (\boldsymbol{\nabla} \times \boldsymbol{A}) + \frac{1}{\mu \sigma} \boldsymbol{\nabla} \times (\boldsymbol{\nabla} \times \boldsymbol{A}) - \boldsymbol{\nabla} (\frac{1}{\mu \sigma} \boldsymbol{\nabla} \cdot \boldsymbol{A}) + \boldsymbol{\nabla} V - \boldsymbol{\nabla} \times \boldsymbol{M} = \mathbf{0} \tag{3.37}$$

and using the curl-curl identity and recalling that $\boldsymbol{\nabla} \cdot \boldsymbol{A} = 0$ then

$$\frac{\partial \boldsymbol{A}}{\partial t} - \boldsymbol{v} \times (\boldsymbol{\nabla} \times \boldsymbol{A}) + \boldsymbol{\nabla}^2 \frac{1}{\mu\sigma} \boldsymbol{A} + \boldsymbol{\nabla} V - \boldsymbol{\nabla} \times \boldsymbol{M} = \boldsymbol{0} \, . \tag{3.38}$$

Using equation (3.3), the G/FEM residual form is

$$\begin{aligned} R^i_{m,k} = & \int_{\mathrm{V}} \left[\left(\rho \frac{\partial \boldsymbol{A}}{\partial t} - \boldsymbol{v} \times (\boldsymbol{\nabla} \times \boldsymbol{A}) + \boldsymbol{\nabla} V - \boldsymbol{\nabla} \times \boldsymbol{M} \right) \cdot \boldsymbol{e}_k \phi^i + \frac{1}{\mu\sigma} \boldsymbol{\nabla} \boldsymbol{A} \cdot \boldsymbol{\nabla} \left(\boldsymbol{e}_k \phi^i \right) \right] \mathrm{dV} \\ & - \int_{\mathrm{S}} \frac{1}{\mu\sigma} (\boldsymbol{n} \cdot \boldsymbol{\nabla} \boldsymbol{A}) \cdot \boldsymbol{e}_k \phi^i \, \mathrm{dS} = 0 \, . \end{aligned} \tag{3.39}$$

In Aria, each term in (3.39) is specified separately as identified in equation (3.40).

$$\begin{aligned} R^i_{m,k} = & \underbrace{\int_{\mathrm{V}} \frac{\partial \boldsymbol{A}}{\partial t} \cdot \boldsymbol{e}_k \phi^i \, \mathrm{dV}}_{\mathrm{MASS}} - \underbrace{\int_{\mathrm{V}} (\boldsymbol{v} \times \boldsymbol{\nabla} \times \boldsymbol{A}) \cdot \boldsymbol{e}_k \phi^i \, \mathrm{dV}}_{\mathrm{ADV}} + \underbrace{\int_{\mathrm{V}} (\boldsymbol{\nabla} V - \boldsymbol{\nabla} \times \boldsymbol{M}) \cdot \boldsymbol{e}_k \phi^i \, \mathrm{dV}}_{\mathrm{SRC}} \\ & + \underbrace{\int_{\mathrm{V}} \frac{1}{\mu\sigma} \boldsymbol{\nabla} \boldsymbol{A} \cdot \boldsymbol{\nabla} \left(\boldsymbol{e}_k \phi^i \right) \, \mathrm{dV}}_{\mathrm{DIFF}} - \int_{\mathrm{S}} \frac{1}{\mu\sigma} (\boldsymbol{n} \cdot \boldsymbol{\nabla} \boldsymbol{A}) \cdot \boldsymbol{e}_k \phi^i \, \mathrm{dS} = 0 \, . \end{aligned} \tag{3.40}$$

Suspension Equation

In treating the suspension as a continuum, we introduce an evolution equation for particle volume fraction, ϕ, as

$$\frac{\partial \phi}{\partial t} + \boldsymbol{v} \cdot \boldsymbol{\nabla} \phi + \boldsymbol{\nabla} \cdot \boldsymbol{N} = 0. \tag{3.41}$$

The particle volume fraction is defined as the total summed volume of particles per volume of the particle medium. (3.41) represents a balance between the stored particles, the convected particle flux, and the diffusive particle flux, N. Several mechanisms which include Brownian motion, sedimentation, hydrodynamic particle interactions, and gradients in suspension viscosity may contribute to the diffusive particle flux. Specification of the appropriate flux model must then be carried out to close the definition of the conservation equation.

Porous Flow Equations

In what follows we use a subscript α to denote a component index and N_α is the total number of components. We use a subscript β to denote a phase and the total number of phases is N_β. The subscript s is used to refer to the solid phase of the porous skeleton; this phase is not included in N_β. Summations over α and β are assumed to range from one to N_α and N_β, respectively, unless otherwise specified.

Generalized Equations

This write up follows, directly, the reports of [8], [9] and [10].

Porosity and Void Ratio

In modeling of porous materials, one often needs to know the relative volume of pore space (V_p) and the solid porous skeleton (V_s). The most common measure of this is the *porosity*, ϕ, which is defined as the ratio of the volume of pore space (V_p) per unit total volume ($V_t = V_p + V_s$), viz.,

$$\phi = \frac{V_p}{V_t} = \frac{V_p}{V_p + V_s}.$$

The *void ratio* (r) is sometimes used as well and is defined as the ratio of the pore volume to the solid volume, viz.,

$$r = \frac{V_p}{V_s}$$

Clearly the porosity and void ratio are related,

$$r = \frac{\phi}{1 - \phi}$$
$$\phi = \frac{r}{1 + r}$$

Component Mass Balances

The mass balance for each component across all phases can be written

$$\frac{\partial d_\alpha}{\partial t} + \boldsymbol{\nabla} \cdot \boldsymbol{F}_\alpha = Q_\alpha \qquad \alpha = 1, \ldots, N_\alpha \tag{3.42}$$

where d_α is mass concentration of component α across all phases, t is time, $\boldsymbol{F}_\alpha$ is the total mass flux of component α relative to a fixed frame of reference and Q_α is the volumetric source of component α.

The mass concentration of component α can be written as

$$d_\alpha = \phi \sum_\beta Y_{\alpha\beta} \rho_\beta S_\beta \tag{3.43}$$

where ϕ is the interstitial volume fraction, a.k.a. the porosity, that ranges from zero to one, Y denotes mass fraction so that $Y_{\alpha\beta}$ is the mass fraction of component α in phase β, ρ_β is mixture density of phase β and S_β is saturation which is defined as the fraction of the interstitial pore space occupied by phase β.

For completeness it is noted here that the mass fractions sum to one in each phase and that the saturations of all phases sum to unity,

$$\sum_{\alpha} Y_{\alpha\beta} = 1 \tag{3.44}$$

$$\sum_{\beta} S_{\beta} = 1 \tag{3.45}$$

The macroscopic mass flux in phase β, $\rho_\beta \boldsymbol{v}_\beta$ is obtained from the extended Darcy's Law,

$$\rho_\beta \boldsymbol{v}_\beta = \boldsymbol{f}_\beta = -\frac{\rho_\beta \kappa_{r\beta}}{\mu_\beta} \boldsymbol{K} \cdot \left(\boldsymbol{\nabla} P_\beta + \rho_\beta \boldsymbol{g} \right) \tag{3.46}$$

where $\kappa_{r\beta}$ is the relative permeability of phase β and $\boldsymbol{K}$ is the permeability tensor which are both intrinsic properties of the porous material and so may vary spatially. The mixture viscosity of phase β is μ_β, P_β is the pressure in phase β and $\boldsymbol{g}$ is the gravitational acceleration.

The component mass flux, $\boldsymbol{F}_\alpha$ can be written as the sum of the component mass fluxes across each phase,

$$\boldsymbol{F}_\alpha = \sum_{\beta} \boldsymbol{F}_{\alpha\beta}. \tag{3.47}$$

Within each phase, the component mass flux can be decomposed in its convective and diffusive components,

$$\boldsymbol{F}_{\alpha\beta} = Y_{\alpha\beta} \boldsymbol{f}_\beta + \boldsymbol{J}_{\alpha\beta} \tag{3.48}$$

where $\boldsymbol{J}_{\alpha\beta}$ is the diffusive flux of component α in phase β. For example, for a gaseous, dilute binary mixture the diffusive flux may be modeled using Fick's Law,

$$\boldsymbol{J}_{\alpha\beta} = -\rho_\beta \mathcal{D}_{\alpha\beta} \boldsymbol{\nabla} Y_{\alpha\beta} \qquad \text{(dilute binary gas mixture)} \tag{3.49}$$

where $\mathcal{D}_{\alpha\beta}$ is the diffusion coefficient of component α in phase β.

Porous Species

The porous species equation is identical to the mass balance equation, with all quantities being molar concentrations. The porous species equation is written as

$$\frac{\partial c_\alpha}{\partial t} + \boldsymbol{\nabla} \cdot \boldsymbol{\Pi}_\alpha = R_\alpha \qquad \alpha = 1, \ldots, N_\alpha, \tag{3.50}$$

where c_α is the molar concentration, defined analogously to (3.43), Π_α is the molar flux, and R_α is the volumetric molar source of component α. The porous species equation exists to facilitate using expressions derived for use with the non porous-flow species equation. An analogous set of terms to those found in the mass balance equation can be derived on a molar concentration basis.

Energy Balance

In order to accommodate multicomponent transport problems we employ a enthalpy based energy conservation law,

$$\frac{\partial e}{\partial t} + \boldsymbol{\nabla} \cdot \boldsymbol{q} = Q_e \tag{3.51}$$

where e is the total internal energy, $\boldsymbol{q}$ is the total heat flux relative to a stationary frame of reference and Q_e is the volumetric source of energy. The total internal energy is written as a sum of the energies of all components across all phases as well as the internal energy of the solid porous structure,

$$e = (1 - \phi)\, \rho_s e_s + \phi \sum_\beta S_\beta \rho_\beta e_\beta \tag{3.52}$$

where ρ_β and e_β are the mixture density and internal energy of phase β. Generally, thermodynamic relations are required to relate these quantities to the component specific quantities.

The total heat flux can be decomposed into different modes of transport. Typically,

$$\boldsymbol{q} = -\lambda_T \boldsymbol{\nabla} T + \sum_\beta \sum_\alpha h_{\alpha\beta} \boldsymbol{F}_{\alpha\beta} \tag{3.53}$$

where λ_T is the effective thermal conductivity, T is the temperature and $h_{\alpha\beta}$ is the partial enthalpy of component α in phase β which is interpreted as the enthalpy of pure component α at the temperature and pressure of phase β. Using equation (3.48), we can expand equation (3.53) such that

$$\begin{aligned} \sum_\alpha h_{\alpha\beta} \boldsymbol{F}_{\alpha\beta} &= \sum_\alpha h_{\alpha\beta} \left(Y_{\alpha\beta} \boldsymbol{f}_\beta + \boldsymbol{J}_{\alpha\beta} \right) \\ &= h_\beta \boldsymbol{f}_\beta + \sum_\alpha h_{\alpha\beta} \boldsymbol{J}_{\alpha\beta} \end{aligned}$$

where $h_\beta = \sum_\alpha h_{\alpha\beta} Y_{\alpha\beta}$ is the mixture enthalpy of phase β. Thus, the total heat flux can be expressed as the sum of conductive, convective and diffusive contributions,

$$\boldsymbol{q} = -\lambda_T \boldsymbol{\nabla} T + \sum_\beta \rho_\beta \boldsymbol{v}_\beta h_\beta + \sum_\beta \sum_\alpha h_{\alpha\beta} \boldsymbol{J}_{\alpha\beta} \tag{3.54}$$

Porous Flow Systems

To facilitate various flavors of single phase and multiphase flow in Aria, the porous flow system model was created to manage expression creation. In addition to the single phase model (*Single Phase, Single Component, Isothermal Flow* and *Single Phase, Single Component, Non-isothermal Flow*), there are several others available including: mixed single phase, *CO2-salt-brine*,

two-phase-immiscible (*Two-Phase Immiscible* and *Pressure-pressure Formulation for Two-phase Immiscible*), *Two-phase Air-Water*, and *Organic Material Decomposition (OMD)*. The basic purpose of the porous flow system model is to create the necessary expressions for each system. There are examples of each flow system in the regression tests, which are the best place to start when developing an input for a similar system. The input syntax for a porous flow system is given in *Material Properties* in addition to the required parameters for each porous flow system.

Single Phase, Single Component, Isothermal Flow

The simplest possible case is for a single phase flow ($N_\beta = 1$, $S = 1$) with a single component ($N_\alpha = 1$, $Y = 1$) at a fixed temperature. This is the single phase porous flow system. For a single component flow, there is no diffusive mass flux so $\boldsymbol{J} = \boldsymbol{0}$ and $\boldsymbol{F} = \boldsymbol{f}$ where

$$\boldsymbol{F} = \boldsymbol{f} = \rho \boldsymbol{v} = -\frac{\rho \kappa_r}{\mu} \boldsymbol{K} \cdot (\boldsymbol{\nabla} P + \rho \boldsymbol{g}) \tag{3.55}$$

In this case, the mass balance equation is all that is needed to solve for the unknown pressure field P.

$$\frac{\partial d}{\partial t} + \boldsymbol{\nabla} \cdot \boldsymbol{f} = Q \tag{3.56}$$

Since the density is most likely a function of temperature, the constant uniform temperature will have to be specified. Finally, in this special case, equation (3.43) reduces to $d = \phi \rho$.

Single Phase, Single Component, Non-isothermal Flow

Building on the simple case developed in the section above (*Single Phase, Single Component, Isothermal Flow*), the next step is to accommodate a non-isothermal flow. Here, we still have a single phase flow ($N_\beta = 1$, $S = 1$) with a single component ($N_\alpha = 1$, $Y = 1$). For a single component flow, there is no diffusive mass flux so $\boldsymbol{J} = \boldsymbol{0}$ and $\boldsymbol{F} = \boldsymbol{f}$ where

$$\boldsymbol{F} = \boldsymbol{f} = \rho \boldsymbol{v} = -\frac{\rho \kappa_r}{\mu} \boldsymbol{K} \cdot (\boldsymbol{\nabla} P + \rho \boldsymbol{g}) \tag{3.57}$$

Again, the mass balance equation is all that is needed to solve for the unknown pressure field P.

$$\frac{\partial d}{\partial t} + \boldsymbol{\nabla} \cdot \boldsymbol{f} = Q \tag{3.58}$$

Again, in this special case, equation (3.43) reduces to $d = \phi \rho$.

In order to solve for the unknown temperature, T, we solve an energy transport equation in the enthalpy form as described in *Energy Balance*. Here, the energy flux in equation (3.53) reduces to

$$\boldsymbol{q} = -\lambda_T \boldsymbol{\nabla} T + \rho \boldsymbol{v} h$$

where h is the pure component enthalpy which may be a function of P and T.

Two-Phase Immiscible

In this section, we define a formulation for two-phase flow of immiscible fluids, that is, without mass transfer between the phases. Common examples include oil/water, under pressure and temperature conditions where phase transitions are negligible, and systems involving non-aqueous phase liquids (NAPL) occurring in subsurface contamination by organic liquids, e.g., solvents, fuels, oils. If we disallow partitioning of components across phases, then we get the following system of mass balance equations for a wetting and non-wetting phase,

$$\frac{\partial}{\partial t}(\rho_w \phi(1 - S_n)) = \boldsymbol{\nabla} \cdot \left(\rho_w \frac{k_{rw}}{\mu_w} \boldsymbol{K} \cdot (\boldsymbol{\nabla} p - \rho_w \boldsymbol{g})\right) + Q_w$$

$$\frac{\partial}{\partial t}(\rho_n \phi S_n) = \boldsymbol{\nabla} \cdot \left(\rho_n \frac{k_{rn}}{\mu_n} \boldsymbol{K} \cdot (\boldsymbol{\nabla} p + \boldsymbol{\nabla} p_c - \rho_n \boldsymbol{g})\right) + Q_n$$

In these equations, the subscript w represents wetting phase properties and the subscript n nonwetting phase properties. We have also incorporated the Darcy flux terms, including pressure and gravitational forces, as well as capillary pressure:

$$p_c = p_n - p_w$$

The symbol p in the equations is used in place of wetting phase pressure, pw. Also, the pore space is assumed saturated with fluids,

$$S_w + S_n = 1$$

This constraint has been incorporated in the wetting phase mass balance. In addition to the density models described in the previous section, models for density and viscosity can be specified as constant, polynomial with respect to other primary or secondary variables, or by tabular functions, or a user-defined plug-in.

Both the capillary pressure and relative permeability are functions of phase saturations. Either of these can be specified by the user in a tabular form, as a function of either phase saturation. A plug-in can also be written by the user for either of these functions. Several commonly used functions are built-in to the code.

Pressure-pressure Formulation for Two-phase Immiscible

The $P_w - S_n$ formulation is generally known to have superior numerical behavior for $S_w \rightarrow 0$, however, the capillary pressure in heterogeneous media creates a problem for a vertex-centered (e.g. FEM) numerical scheme. In FEM, because a node straddles a material interface, which separates two materials with different texture and therefore different wetting properties, the capillary pressure is continuous across the interface, but the saturation is not. So, a (P_w,:math:S_n) formulation in FEM would require dealing with discontinuous DOF for S_n. One alternative is to instead solve for two pressures. Two pairs of pressure-based DOFs available in Aria include

(P_w, P_c) and (P_w, P_n). (Note that in single phase regions, where $S_w = 0$ or 1, then P_w or P_n are undefined. The formulation should work for the thermal battery problem so long as $S_w > 0$.

The following system of mass balance equations for a wetting and non-wetting phase describes an immiscible flow,

$$\frac{\partial}{\partial t}(\rho_w \phi S_w) = \boldsymbol{\nabla} \cdot \left(\rho_w \frac{k_{rw}}{\mu_w} \boldsymbol{K} \cdot (\boldsymbol{\nabla} P_w - \rho_w \boldsymbol{g}) \right) + Q_w$$

$$\frac{\partial}{\partial t}(\rho_n \phi S_n) = \boldsymbol{\nabla} \cdot \left(\rho_n \frac{k_{rn}}{\mu_n} \boldsymbol{K} \cdot (\boldsymbol{\nabla} P_w + \boldsymbol{\nabla} P_c - \rho_n \boldsymbol{g}) \right) + Q_n$$

In these equations, the subscript w represents wetting phase properties and the subscript n non-wetting phase properties. We have also incorporated the Darcy flux terms, including pressure and gravitational forces, as well as capillary pressure:

$$p_c = p_n - p_w$$

Also, the pore space is assumed saturated with fluids,

$$S_w + S_n = 1$$

The current implementation in Aria uses a van Genuchten function to model the relationship between the capillary pressure and the saturation. This relationship is inverted for saturation given the capillary pressure in the pressure-pressure formulation.

Two-phase Air-Water

This model describes the nonisothermal two-component, two-phase transport of water and air in a porous medium. The air is treated as a non-condensible gas, but it can dissolve in the liquid phase. The water can exist as water vapor or liquid. The model allows the fluids to partition between the two phases, depending on pressure and temperature conditions. Two single phase states (all liquid or all gas) and one two-phase state are allowed.

The component mass balance equations describing two-phase transport of water, air and energy are:

$$\frac{\partial}{\partial t}(\phi(S_l Y_{wl} \rho_l + S_g Y_{wg} \rho_g)) + \boldsymbol{\nabla} \cdot (Y_{wl} \rho_l \boldsymbol{v}_l + Y_{wg} \rho_g \boldsymbol{v}_g + \boldsymbol{J}_{wg}) + Q_w = 0$$

$$\frac{\partial}{\partial t}(\phi(S_l Y_{al} \rho_l + S_g Y_{ag} \rho_g)) + \boldsymbol{\nabla} \cdot (Y_{al} \rho_l \boldsymbol{v}_l + Y_{ag} \rho_g \boldsymbol{v}_g + \boldsymbol{J}_{ag}) + Q_a = 0$$

$$\begin{aligned} \frac{\partial}{\partial t}\left[(1-\phi)\rho_s e_s + \phi(\rho_l S_l e_l + \rho_g S_g e_g)\right] \\ + \boldsymbol{\nabla} \cdot (-\lambda_T \boldsymbol{\nabla} T + \rho_l \boldsymbol{v}_l h_l + \rho_g \boldsymbol{v}_g h_g + h_{wg} \boldsymbol{J}_{wg} + h_{ag} \boldsymbol{J}_{ag}) \\ + Q_e = 0 \end{aligned}$$

where

$$\boldsymbol{J}_{ag} = -\rho_g D_{wa} \boldsymbol{\nabla} Y_{ag}$$

is the gas-phase binary diffusion flux for water vapor through air. In these equations, subscript l refers to the liquid phase and g to the gas phase. In addition to previously defined variables: $Y_{\alpha\beta}$ is the mass fraction of component α (w for water, a for air) in phase β, S_β is the saturation of phase β, ρ_β is the phase density, e_β is the internal energy of phase β, λ_T is the effective thermal heat conduction, h_β is the enthalpy of phase β and $h_{\alpha g}$ is the component enthalpy in the gas phase.

The liquid and gas phase Darcy velocities are

$$\boldsymbol{v}_l = -\frac{k_{rl}}{\mu_l} \boldsymbol{K} \cdot (\boldsymbol{\nabla} P_l + \rho_l \boldsymbol{g})$$

$$\boldsymbol{v}_g = -\frac{k_{rg}}{\mu_g} \boldsymbol{K} \cdot (\boldsymbol{\nabla} P_g + \rho_g \boldsymbol{g})$$

In addition, the following constraints hold:

$$S_l + S_g = 1 \; ; \quad Y_{w\beta} + Y_{a\beta} = 1 \text{ , for } \beta = l \text{ or } g$$

CO2-salt-brine

The CO2-salt-brine system is intended to model the interaction of CO2 gas, solid salt, and liquid brine. In this system there are three mass balance equations that must be solved for the saturations of each. This formulation describes a 3-component 2-phase flow for the water-CO2-NaCl system. The two phases are brine, denoted by subscript w, a liquid phase of water with salt and CO2 dissolved in it, and "gas" denoted by subscript g, a CO2-rich phase, with some water dissolved in it, but no salt. The formulation also deals with precipitation of salt from the liquid phase.

To deal with the precipitation of NaCl from the liquid to form solid salt, the fluid system is treated as a three-phase system, with two flowing phases composed of the brine and CO2 phases, and one non-flowing solid phase, which is the solid salt precipitate. Obviously, the formulation holds for other chloride salts, e.g. CaCl, KCl.

$$\frac{\partial}{\partial t}\phi(S_w \rho_w x_w + S_g \rho_g y_w) + \boldsymbol{\nabla} \cdot (\rho_w x_w \boldsymbol{v}_w + \rho_g y_w \boldsymbol{v}_g + \boldsymbol{J}_w^{H2O} + \boldsymbol{J}_g^{H2O}) = Q^{H2O}$$

$$\frac{\partial}{\partial t}\phi(S_w \rho_w x_{CO2} + S_g \rho_g y_{CO2}) + \boldsymbol{\nabla} \cdot (\rho_w x_{CO2} \boldsymbol{v}_w + \rho_g y_{CO2} \boldsymbol{v}_g + \boldsymbol{J}_w^{CO2} + \boldsymbol{J}_g^{CO2} = Q^{CO2}$$

$$\frac{\partial}{\partial t}\phi(S_w \rho_w x_{NaCl} + S_g \rho_s) + \boldsymbol{\nabla} \cdot (\rho_w x_{NaCl} \boldsymbol{v}_w + \boldsymbol{J}_w^{NaCl}) = Q^{NaCl}$$

In the equations above, we have used the following nomenclature, S_s is the saturation of solid salt precipitate, S_w is the brine phase saturation, S_g is the CO2-rich phase saturation, x_w is the mass fraction of water in the brine phase, x_{CO2} is the mass fraction of CO2 in the brine phase, x_{NaCl} is the mass fraction of salt dissolved in the brine phase, y_{CO2} is the mass fraction of CO2 in the

CO2-rich ("gas") phase, y_w is the mass fraction of water in the gas phase, $\rho_s(T, p)$ is the density of solid precipitate, and $\boldsymbol{J}_\beta^\alpha$ is the diffusion flux of component α in phase β which can also include dispersion. The subscript 'w' refers to the brine phase, 'g' refers to the CO2 phase and 's' refers to the solid salt phase.

This model treats all the separate-phase CO2 as a single CO2-rich phase. The phase diagram for the CO2-H20 system displays a saturation line where liquid water, liquid CO2 and a vapor phase can co-exist. The saturated water vapor pressure on this line is small, so the vapor phase is mostly CO2.

$$S_w + S_g + S_s = 1 \; ; \quad x_w + x_{CO2} + X_{NaCl} = 1 \; ; \quad y_w + y_{CO2} + y_{NaCl} = 1$$

We also have the constraint on the over-all composition, which can replace either of the last two constraints above,

$$z_w + z_{CO2} + z_{NaCl} = 1$$

To solve for the pressure, in all compositional situations, we can solve a pressure equation, obtained by summing all component balances. Since the resulting equation is not independent, it replaces one of the three mass balance equations presented earlier.

$$\frac{\partial}{\partial t}\phi(S_w\rho_w + S_g\rho_g + S_s\rho_s) + \boldsymbol{\nabla} \cdot (\rho_w \boldsymbol{v}_w + \rho_g \boldsymbol{v}_g) + Q^{H2O} + Q^{CO2} + Q^{NaCl} = 0$$

In the numerical implementation, this equation could be obtained after all BCs have been applied to the discrete equations, by summing the three residual equations together.

Recall, the primary variables for this system are defined as a set of variables that allow one to compute all the remaining (secondary) variables appearing in the mass and energy balance equations. For this problem, the primary solution vector (assuming we will also solve an energy balance for T) is $[P, z_{CO2}, \rho_{NaCl}, T]$, where

$$z_{CO2} = \frac{S_w\rho_w x_{CO2} + S_g\rho_g y_{CO2}}{S_w\rho_w + S_g\rho_g} = (1-\nu)x_{CO2} + \nu y_{CO2}$$

is the over-all mass fraction of CO2 in the flowing phases. We have introduced the fraction of gas, defined as,

$$\nu = \frac{\rho_g S_g}{\rho_w S_w + \rho_g S_g}$$

Similarly, the overall density of salt is defined by,

$$\rho_{NaCl} = \rho_w S_w x_{NaCl} + \rho_s S_s = (1-\nu)x_{NaCl}F + \rho_s S_s$$

where we have introduced, for later convenience, the following mass density,

$$F = \rho_w S_w + \rho_g S_g$$

Note that the accumulation term in the CO2 balance can be written as $\phi F z_{CO2}$.

In this formulation, P is solved for from the pressure equation, z_{CO2} from the CO2 mass balance equation, ρ_{NaCl} from the NaCl mass balance and T from the energy balance equation. We assume the phase equilibrium for this system is done ahead of time.

The thermodynamics algorithm for this system is still in development and has not yet been completed.

Organic Material Decomposition (OMD)

Organic material decomposition (OMD) models currently define two-phase (solid and gas) problems in which the solid (or porous) species react to produce additional gas species that pressurize a system and volumetric heat transfer between phases. The notation used above does not include the porous solid as a phase, but since for OMD we are solving equations for the solid phase including reactions, we set $N_\beta = 2$, with it being understood that one of these phases is the solid phase. OMD problems will extend to include a liquid phase in the future. The OMD equations described here are based on the combustion model [11]. Further details about coupling boundary conditions between fluid-only and porous regions are provided in *Coupling BCs for Organic Material Decomposition*. There are separate mass balance equations for the solid phase and gas phase. The solid phase mass balance equation is used to solve for solid phase density:

$$\underbrace{\frac{\partial \bar{\rho}}{\partial t}}_{MASS} = \underbrace{-\dot{\omega}'''_{f,g}}_{SRC}. \tag{3.59}$$

where $\bar{\rho}$ is the solid phase density and $\dot{\omega}'''_{f,g}$ is the rate of formation of gases due to heterogenous reaction kinetics. The mass balance equation for the gas phase solves for the pressure (from the gas phase density via the ideal gas law),

$$\underbrace{\frac{\partial(\psi\rho_g)}{\partial t}}_{MASS} + \underbrace{\frac{\partial(\rho_g u_{j,g})}{\partial x_j}}_{ADV} = \underbrace{\dot{\omega}'''_{f,g}}_{SRC}, \tag{3.60}$$

where ρ_g is the gas-phase density and ψ is the mixture-averaged solid-phase porosity. The porosity is the void space not taken up by the volume fractions of each solid phase species, typically defined as a function of the volume fractions of each solid species,

$$\psi = 1 - \sum_\alpha X_\alpha.$$

The volume fraction of each solid species is calculated as

$$X_\alpha = \frac{\bar{\rho} Y_\alpha}{\rho_{s0,\alpha}},$$

where $\rho_{s0,\alpha}$ is the solid (non-porous) density for each species. In equation (3.60), $u_{j,g}$ is the gas-phase velocity and gets computed from Darcy's approximation as

$$\psi v_{j,g} = u_{j,g} = -\frac{\bar{K_{ij}}}{\mu_g}\left(\frac{\partial p_g}{\partial x_i} + \rho_g g_i\right), \tag{3.61}$$

where $\bar{K_{ij}}$ is the mixture-averaged solid-phase permeability tensor defined as the normalized volume average of the solid phase species' permeabilities,

$$\bar{K_{ij}} = \frac{\sum_\alpha X_\alpha K_{\alpha j}}{\sum_\alpha X_\alpha}$$

where

$$K_{\alpha j} = K_{ref}\left(\frac{T}{T_{ref}}\right)^n \delta_{\alpha j}$$

The intrinsic permeability is the product of the intrinsic permeability scaling and the intrinsic permeability. The reason it is separated in this way is that scalar models have access to many generic functions, whereas tensor entries can only be constants. A more complex function for the intrinsic permeability tensor would have to be written as a user subroutine.

In equation (3.61) the parameter μ_g is the gas-phase viscosity and g_j is the gravity vector. The ideal gas law gets used for the gas density in the pressure form of equation (3.60)

$$\rho_g = \frac{P\bar{M}}{RT_g}.$$

Therefore the actual equation for pressure being solved here is

$$\frac{\partial}{\partial t}\left(\frac{p_g \bar{M}\psi}{RT_g}\right) + \frac{\partial}{\partial x_j}\left[\frac{p_g \bar{M}}{RT_g}\frac{\bar{K_{\alpha j}}}{\mu_g}\left(\frac{\partial p_g}{\partial x_j} + \frac{p_g \bar{M}}{RT_g} g_j\right)\right] = 0. \tag{3.62}$$

where $\bar{M}$ is the mass averaged molecular weight,

$$\bar{M} = \sum_\alpha Y_\alpha M_\alpha,$$

where Y_α are the gas phase mass fractions ans M_α are the molecular weights of the gaseous species.

The solid phase species equations solve for the solid phase mass fractions, $Y_{\alpha,s}$, of species α

$$\underbrace{\frac{\partial(\bar{\rho}Y_\alpha)}{\partial t}}_{MASS} = \underbrace{\left(\dot{\omega}'''_{f,\alpha} - \dot{\omega}'''_{d,\alpha}\right)}_{SRC}. \tag{3.63}$$

where $\dot{\omega}'''_{f,\alpha}$ is the formation rate of species α and $\dot{\omega}'''_{d,\alpha}$ is its destruction rate. The gas-phase species equations solve for the gas-phase mass fraction $Y_{\alpha,g}$ of species α

$$\underbrace{\frac{\partial(\psi\rho_g Y_{\alpha,g})}{\partial t}}_{MASS} + \underbrace{\frac{\partial(\rho_g u_{j,g} Y_{\alpha,g})}{\partial x_j}}_{ADV} + \underbrace{\frac{\partial q_{\alpha j}^{Y,g}}{\partial x_j}}_{DIFF} = \underbrace{\dot{\omega_{\alpha,g}}}_{SRC} \tag{3.64}$$

No source terms are included in this model. The gas-phase species diffusion flux vector $q_{\alpha j}^{Y,g}$ is modeled as

$$q_{\alpha j}^{Y,g} = -\psi \rho_g D_{\alpha,g} \frac{\partial Y_{\alpha,g}}{\partial x_j},$$

where $D_{\alpha,g}$ is the gas-phase mass diffusivity for species α.

Enthalpy equations are used to solve for temperature, and OMD type problems allow for different temperature fields in the gas and solid phases, so these are described separately here.

The solid phase enthalpy equation for the solid phase temperature $\bar{T}$ is given by

$$\underbrace{\frac{\partial(\bar{\rho} c_p \bar{T})}{\partial t}}_{MASS} = \underbrace{-\frac{\partial q_j^h}{\partial x_j}}_{DIFF} + \underbrace{h_{cv}\left(T_g - \bar{T}\right)}_{SRC} \tag{3.65}$$

There is no advection term here since the solid phase is stationary. The solid phase energy diffusive flux vector $q_j^{h,g}$ is given by Fourier's law,

$$q_j^{h,g} = -\kappa \frac{\partial \bar{T}}{\partial x_j},$$

where κ is the thermal conductivity.

The gas-phase enthalpy equation within a porous region, to be solved for the gas-phase enthalpy (and corresponding temperature) $h_g = c_p T_g$, is

$$\underbrace{\frac{\partial(\psi \rho_g h_g)}{\partial t}}_{MASS} + \underbrace{\frac{\partial(\rho_g u_{j,g} h_g)}{\partial x_j}}_{ADV} = \underbrace{-\frac{\partial q_j^{h,g}}{\partial x_j}}_{DIFF} + \underbrace{\left(\frac{\partial \psi p_g}{\partial t}\right)}_{SRC} + \underbrace{h_{cv}\left(\bar{T} - T_g\right)}_{SRC} \tag{3.66}$$

where h_g is the mixture-averaged gas-phase enthalpy, h_{cv} is the volumetric heat transfer coefficient which gets set outside of the solid/gas phases of the material definition, $\bar{T}$ is the porous solid-phase temperature and T_g is the gas phase temperature. The gas-phase energy diffusive flux vector $q_j^{h,g}$ is modeled as

$$q_j^{h,g} = -\psi \rho_g D_g \frac{\partial h_g}{\partial x_j},$$

where D_g is the mixture-averaged gas-phase mass diffusivity. The compressible pressure source term for the enthalpy equation has been derived in the context of Darcy flow. The volumetric transfer source accounts for the heat transfer between the solid and gas phases.

Porosity Models

There are several models for porosity in the porous flow capabilities in Aria. The following list provides a brief description for each.

Mesh Deforming

In the mesh (or solid) deforming porosity model, the expression for the porosity is given as

$$\phi = 1.0 - \frac{b(1.0 - \phi_0)}{\det(\mathrm{F})}$$

where ϕ_0 is the initial porosity, b is the Biot coefficient, and F is the mesh deformation gradient.

Rock Compressible

For the Rock Compressible porosity model, the porosity is given as

$$\phi = \phi_0(1.0 + Cr(p - p_{ref}))$$

where again, ϕ_0 is the initial porosity, Cr is a rock compressibility coefficient, p is the pore pressure, and p_{ref} is a reference pore pressure defined in the input parameters.

Coussy

For the Coussy porosity model, the expression for porosity is

$$\phi = \phi_0 + \frac{1}{M}(p - p_{ref}) + b\epsilon_u$$

where M represents the formation compressibility, $\epsilon_u = \mathrm{div}(\boldsymbol{u})$ is the divergence of the solid deformation vector, b is the Biot coefficient, p is the pore pressure, and p_{ref} is a reference pore pressure.

Brinkman Momentum

The Brinkman (Brinkman-Forchheimer) momentum equation is a porous media specialization of the fluid momentum equation (3.21) given by

$$\frac{\partial}{\partial t}\left(\frac{\rho}{\varphi}\boldsymbol{v}\right) + \frac{\rho}{\varphi^2}\boldsymbol{v}\cdot\boldsymbol{\nabla}\boldsymbol{v} - \boldsymbol{\nabla}\cdot\boldsymbol{T} + \left[\frac{\rho\hat{c}}{\sqrt{k}}\|\boldsymbol{v}\| + \frac{\mu}{k}\right]\boldsymbol{v} - g = \boldsymbol{0} \tag{3.67}$$

where φ is the matrix porosity, $\boldsymbol{T}$ is the fluid stress tensor $\boldsymbol{g}$ is a body force and μ is termed the flowing liquid viscosity. The bracketed terms are often referred to as the Forchheimer or inertial

flow contribution and the Darcy term respectively. In the limit of low velocities the equation reduces to a balance of pressure and a scaled velocity and Darcy's law for a single component fluid is recovered [12]. This equation is most useful in cases where a clear fluid domain is bounded by a porous media. In most applications of this equation the hydrostatic contribution will be negligible but is retained here for completeness. We construct the G/FEM residual form of (3.67) by contracting with the unit coordinate vector in the k-direction, $\boldsymbol{e}_k$, multiplying by the weight function ϕ^i and integrating over the volume. Using the vector identity $(\boldsymbol{\nabla}\cdot\boldsymbol{T})\cdot\boldsymbol{e}_k\phi^i = \boldsymbol{\nabla}\cdot(\boldsymbol{T}\cdot\boldsymbol{e}_k\phi^i) - \boldsymbol{T}^t : \boldsymbol{\nabla}(\boldsymbol{e}_k\phi^i)$ and integrating by parts gives

$$\begin{aligned} R^i_{m,k} = &\int_{\mathrm{V}} \left[\frac{\partial}{\partial t}\left((\frac{\rho}{\varphi}\boldsymbol{v}) + \frac{\rho}{\varphi^2}\boldsymbol{v}\cdot\boldsymbol{\nabla}\boldsymbol{v} + \left[\frac{\rho\hat{c}}{\sqrt{k}}\|\boldsymbol{v}\| + \frac{\mu}{k}\right] - g\right)\cdot\boldsymbol{e}_k\phi^i + \boldsymbol{T}^t : \boldsymbol{\nabla}\left(\boldsymbol{e}_k\phi^i\right)\right] \mathrm{dV} \\ &- \int_{\mathrm{S}} \boldsymbol{n}\cdot\boldsymbol{T}\cdot\boldsymbol{e}_k\phi^i \,\mathrm{dS} = 0 \end{aligned} \tag{3.68}$$

Here we note that the viscosity associated with the stress tensor $\boldsymbol{T}$ is not the viscosity of the fluid but instead a so-called Brinkman viscosity μ_B. In Aria, each term in (3.68) is specified separately as identified in equation (3.18).

$$\begin{aligned} R^i_{m,k} = &\underbrace{\int_{\mathrm{V}} \frac{\partial}{\partial t}(\frac{\rho}{\varphi}\boldsymbol{v})\cdot\boldsymbol{e}_k\phi^i \,\mathrm{dV}}_{\text{MASS}} + \underbrace{\int_{\mathrm{V}} \frac{\rho}{\varphi^2}\boldsymbol{v}\cdot\boldsymbol{\nabla}\boldsymbol{v}\cdot\boldsymbol{e}_k\phi^i \,\mathrm{dV}}_{\text{ADV}} - \underbrace{\int_{\mathrm{V}} \boldsymbol{g} - \left[\frac{\rho\hat{c}}{\sqrt{k}}\|\boldsymbol{v}\| + \frac{\mu}{k}\right]\boldsymbol{v}\cdot\boldsymbol{e}_k\phi^i \,\mathrm{dV}}_{\text{SRC}} \\ &+ \underbrace{\int_{\mathrm{V}} \boldsymbol{T}^t : \boldsymbol{\nabla}\left(\boldsymbol{e}_k\phi^i\right) \,\mathrm{dV}}_{\text{DIFF}} - \int_{\mathrm{S}} \boldsymbol{n}\cdot\boldsymbol{T}\cdot\boldsymbol{e}_k\phi^i \,\mathrm{dS} = 0 \end{aligned} \tag{3.69}$$

Lubrication Equation

Reynolds' lubrication equation is a reduced-order model for fluid flow in thin, confined regions. This equation is commonly used for manufacturing applications, such as bearings, coating processes, and nanomanufacturing processes. Lubricating flows also occur in high-speed machinery, such as pumps. It is only defined for reduced-order elements, such as shells.

The lubrication equation implemented here is given by

$$-\frac{\partial h}{\partial t} + \boldsymbol{\nabla}\cdot\boldsymbol{q} + \boldsymbol{B}_u\cdot\boldsymbol{\nabla}h_u - \boldsymbol{B}_l\cdot\boldsymbol{\nabla}h_l = 0 \tag{3.70}$$

where $\boldsymbol{q}$ is the lubrication flow rate, given by

$$\boldsymbol{q} = -\frac{h^3}{k\mu}\boldsymbol{\nabla}p_{\mathrm{lub}} + \frac{h}{2}(\boldsymbol{B}_u + \boldsymbol{B}_l). \tag{3.71}$$

In these equations, the independent variable is the lubrication pressure, p_{lub}. Other variables are the lubrication region height (thickness) h, velocity of the upper confining surface $\boldsymbol{B}_u$, velocity of

the lower confining surface $\boldsymbol{B}_l$, the upper wall height h_u, the lower wall height h_l, turbulent parameter k, and viscosity μ. The lubrication height is defined as $h = h_u - h_l + \boldsymbol{n} \cdot \boldsymbol{d}$, where $\boldsymbol{n}$ is the shell normal vector and $\boldsymbol{d}$ is the mesh displacement. This equation is detailed in [13].

Using equation (3.70), the G/FEM residual form is

$$R_V^i = \underbrace{-\int_V \frac{\partial h}{\partial t}\phi^i \, \mathrm{dV}}_{\text{MASS}} + \underbrace{\int_V \boldsymbol{q} \cdot \boldsymbol{\nabla}\phi^i \, \mathrm{dV}}_{\text{DIFF}} + \underbrace{\int_V (\boldsymbol{B}_u \cdot \boldsymbol{\nabla} h_u - \boldsymbol{B}_l \cdot \boldsymbol{\nabla} h_l)\phi^i \, \mathrm{dV}}_{\text{BOUND}} + \int_S \boldsymbol{n} \cdot \boldsymbol{q}\phi^i \, \mathrm{dS} = 0 \tag{3.72}$$

Stress Tensor Projection Equation

A projection equation is defined as an equation where a derived quantity at the interior Gauss points is evaluated and projected to be a solution unknown at the nodal points. The stress tensor projection equation projects the momentum stresses, tau, without the pressure term, to the nodal points. Projecting the momentum stress smooths out the momentum stress tensor and allows for a dot product to be carried out on the projected field, which is needed for least squares stabilization schemes. The solution variable, τ_{ab}, is calculated from (3.73).

$$R_V^i = \underbrace{-\int_V (\tau_{ab} - src_\tau_{ab})\phi^i \, \mathrm{dV}}_{\text{DEF}} = 0 \tag{3.73}$$

τ_{ab} is a tensor variable. For 2D problems, ab stands for `XX`, `XY`, `YX`, and `YY`. For 3D problems ab stands for `XX`, `XY`, `XZ`, `YX`, `YY`, `YZ`, `ZX`, `ZY`, and `ZZ`. The source term in the equation $src_{\tau ab}$ refers to the momentum stress without the pressure diagonal term.

$$\begin{aligned}
src_\tau_{xx} &= 2\mu\frac{du}{dx} - \frac{2}{3}(\mu - \lambda)(\boldsymbol{\nabla} \cdot \boldsymbol{v}) \\
src_\tau_{yy} &= 2\mu\frac{dv}{dy} - \frac{2}{3}(\mu - \lambda)(\boldsymbol{\nabla} \cdot \boldsymbol{v}) \\
src_\tau_{zz} &= 2\mu\frac{dw}{dz} - \frac{2}{3}(\mu - \lambda)(\boldsymbol{\nabla} \cdot \boldsymbol{v}) \\
src_\tau_{xy} &= src_\tau_{yx} = \mu(\frac{du}{dy} + \frac{dv}{dx}) \\
src_\tau_{xz} &= src_\tau_{zx} = \mu(\frac{du}{dz} + \frac{dw}{dx}) \\
src_\tau_{yz} &= src_\tau_{zy} = \mu(\frac{dv}{dz} + \frac{dw}{dy})
\end{aligned}$$

Potential Projection Equation

This equation is used to create a potential field which minimizes difference between the potential, ψ, and the potential source, $\mathbf{S}$, in the least-squares sense

$$(\boldsymbol{\nabla}\psi - \mathbf{S})^2 = 0. \tag{3.74}$$

This equation was derived to create a hydrostatic potential field, so the most common usage is where $\mathbf{S} = \rho_s \mathbf{g}$.

Equation (3.74) is integrated by parts, resulting in the residual

$$R^i = \underbrace{\int_{\mathrm{V}} \boldsymbol{\nabla}\psi \cdot \boldsymbol{\nabla}(\boldsymbol{e}_k \phi^i)\, \mathrm{dV}}_{\mathrm{DEF}} - \underbrace{\int_{\mathrm{V}} \mathbf{S} \cdot \boldsymbol{\nabla}(\boldsymbol{e}_k \phi^i)\, \mathrm{dV}}_{\mathrm{SRC}} \tag{3.75}$$

Simplified Spherical Harmonics Equations

Radiative transport, which is another mode of heat transfer, describes the spatial variation of radiative intensity corresponding to a given direction and at a given wavelength within a radiatively participating medium. This is governed by the Boltzmann Equation or Radiative Transport Equation (RTE) which represents a balance between absorption, emission and scattering. For the purpose of thermal analysis, we make certain assumptions that simplify the RTE. These are

Gray Media
: The optical properties of the material medium are not a function of wavelength i.e., no spectral banding.

Steady State
: Energy transport is propagated by photons which travel at the speed of light. This means that there are no transient effects relative to the time scale of other physics.

Isotropic Scattering
: Since the scattering phase function is rarely known, we simply assume isotropic scattering where the probability of intensity being propagated in a given direction has equal probability of being scattered in another direction.

Given the above assumptions, we may write the RTE as

$$\Omega \cdot \boldsymbol{\nabla} I\left(\Omega\right) + \left(\sigma_A + \sigma_S\right) I\left(\Omega\right) = \sigma_A I_b + \frac{\sigma_S}{4\pi} G, \tag{3.76}$$

where σ_A is the absorption coefficient, σ_S is the scattering coefficient, $I(\Omega)$ is the intensity along the direction Ω, T is the temperature and the angle-integrated intensity is $G = \int_{4\pi} I\left(\Omega\right) d\Omega$. The

black body radiation, I_b is defined by $\frac{\sigma T^4}{\pi}$. Note that for situations in which the scattering coefficient is zero, the RTE reduces to a set of linear, decoupled equations for each intensity to be solved.

Discrete Ordinate (DO) methods such as S_N have been developed to help solve this set of coupled equations by selecting a set of prescribed directions Ω_k, solving for the discrete intensities I_k and combining these intensities to represent the radiative field. These methods suffer from high computational cost due to the fact that a large number of ordinate directions is required to adequately resolve the intensity field and avoid ray effects. For a given quadrature order N, the number of ordinate directions scales as N^2.

Beta Capability
SP_N is not well tested and should not be considered a production capability.

The simplified spherical harmonics (SP_N) approximation to the radiative transport equation (RTE) was first proposed by Gelbard [14] for reactor analysis in the early 1960s. The initial derivation involved replacing the spatial derivatives in the 1D spherical harmonics (P_N) approximation with their 3D analogs. It was later shown that a more rigorous theoretical basis was possible and that the SP_N equations may be derived as either an asymptotic correction to the diffusion limit or from the use of certain trial functions in the self-adjoint variational characterization of the even-parity form of the RTE. There are a number of equivalent forms for the SP_N equations in the literature. We choose to use the canonical form which is derived from the 1D even-parity discrete ordinates equations.

$$-\boldsymbol{\nabla}\cdot\left(\frac{\mu_n^2}{\sigma_T}\boldsymbol{\nabla}I_n\right)+\sigma_T I_n = 4\pi\sigma_S\sum_{m=1}^{\frac{N+1}{2}} w_m I_m + \sigma_A I_b \tag{3.77}$$

where σ_T is the extinction coefficient, $\sigma_T = \sigma_A + \sigma_S$, μ_n is the nth quadrature point in a (N+1)-point Gauss set on [-1, 1], I_n is the angular intensity at quadrature point n, and w_n is the n^{th} quadrature weight.

The appropriate boundary conditions for the canonical SPn equations are derived from the boundary conditions for the 1D even-parity discrete ordinates equations. We consider a Mark boundary condition of the 1st order intensity BC

$$I_n = \epsilon I_b + (1-\epsilon)\sum_{\vec{n}\cdot\vec{\Omega}_j<0} I_j w_j \left|\vec{n}\cdot\vec{\Omega}_j\right|$$

such that in canonical form, it yields

$$-\frac{\mu_n}{\sigma_T}\boldsymbol{\nabla}I_n\cdot\vec{n} = \frac{\epsilon}{2-\epsilon}\left(I_n - I_b\right) + \frac{1-\epsilon}{2-\epsilon}\left[\frac{\sum_k\left(I_k - \frac{\mu_k}{\sigma_T}\boldsymbol{\nabla}I_k\cdot\vec{n}\right)\mu_k w_k}{\sum\mu_k w_k} - I_n + \frac{\mu_n}{\sigma_T}\boldsymbol{\nabla}I_n\cdot\vec{n}\right] \tag{3.78}$$

where ϵ is the emissivity and $\vec{n}$ is the surface normal.

The SP_N equations are solved for the unknown intensities to provide an approximate solution to the RTE. The angle-integrated intensity is then found and used in the source term for the material energy equation. It is to be noted from the form of the SP_N equations that this cannot be used in vacuum or near-vacuum situations where σ_T 0. Also analysis by Zheng et al. shows that the existence of a unique solution may be proved when absorption effects are non-negligible and the geometry is small.

Advective Bar Equation

The advective bar block is used to solve for 1-D temperature, velocity and (optionally) pressure to be used for a convective heat transfer boundary condition on a surface. These variables only vary along the length of the bar, and assume a constant value in the cross sectional area for each segment.

Advective Bar Velocity

The advective bar assumes a constant mass flow rate over the length of the bar. Typically, the mass flow rate is specified in the advective bar command block. The bar velocity $\boldsymbol{v}_{bar}$ is then computed as:

$$\boldsymbol{v}_{bar} = \frac{\dot{m}}{\rho A}\hat{\boldsymbol{s}} \tag{3.79}$$

Where $\hat{\boldsymbol{s}}$ is the bar coordinate direction, $\dot{m}$ is the mass flow rate, ρ is the density and A is the cross section area. The bar velocity is then used to compute a Nusselt number Nu (details of declaring this can be found in *Correlation Heat Transfer Coefficient*). The heat transfer correlation coefficient h is then computed as

$$h = \frac{K}{D_h}\mathrm{Nu} \tag{3.80}$$

Where D_h and K are the scalar thermal conductivity and characteristic length for the simulation, which MUST be specified in the heat transfer correlation coefficient command block. A convective heat flux (q_{conv}) can then be applied to the bar

$$q_{conv} = h(T_{bar} - T) \tag{3.81}$$

Where T_{bar} and T are the temperature of the bar and the coupled surface, respectively. Note that the flux applied to the bar is equal and opposite to the flux that will be received by the coupled, opposing domain.

Advective Bar Energy

The energy equation for the 1-D bar block is given as follows:

$$\rho C_p \frac{\partial T_{bar}}{\partial t} + \rho C_p \boldsymbol{v}_{bar} \cdot \boldsymbol{\nabla} T_{bar} = -\boldsymbol{\nabla} \cdot \boldsymbol{q_{bar}} + H_{V,bar}. \tag{3.82}$$

Taking a standard finite element formulation, the G/FEM residual form is:

$$\begin{aligned} R_T^i = & \underbrace{\int_V \rho C_p \frac{\partial T_{bar}}{\partial t} \phi^i \, \mathrm{dV}}_{\text{MASS}} + \underbrace{\int_V \rho C_p \boldsymbol{v}_{bar} \cdot \boldsymbol{\nabla} T_{bar} \phi^i \, \mathrm{dV}}_{\text{ADV}} - \underbrace{\int_V H_{V,bar} \phi^i \, \mathrm{dV}}_{\text{SRC}} \\ & \underbrace{- \int_V \boldsymbol{\nabla} \phi^i \cdot \boldsymbol{q_{bar}} \, \mathrm{dV}}_{\text{DIFF}} + \int_S q_{conv} \phi^i \, \mathrm{dS} = 0 \end{aligned} \tag{3.83}$$

where q_{conv} includes any surface heat transfer occurring between the bar and the coupled surfaces (Eq. (3.81)). The diffusion term in Eq. (3.83) ends up being an axial diffusion term; for advection dominant flows, this diffusion term may be dropped and SUPG stabilization is recommended.

Beta Capability
BAR MOMENTUM EQUATION is a beta feature and requires use of the –beta flag. This capability is not thoroughly tested and should be used with caution. Refer to *Current Discrepancies with Advective Bars* for proper usage.

Advective Bar Pressure

The velocity is assumed to be known from Eq. (3.79). Assuming steady state flow, a Newtonian fluid and continuity, the G/FEM residual form of the bar pressure P_{bar} is:

$$R_P^i = \int_V \phi^i \hat{\boldsymbol{s}} \cdot (\rho \boldsymbol{v}_{bar} \cdot \boldsymbol{\nabla} \boldsymbol{v}_{bar} - \rho \boldsymbol{g} + \nabla P_{bar}) \, \mathrm{dV} + \int_S \tau_w \phi^i \, \mathrm{dS} = \boldsymbol{0} \tag{3.84}$$

Where τ_w is the wall shear stress. This shear wall shear stress resolves any head loss due to friction along the length of the bar and the connected surface. It should be noted that within Aria, the surface integral of τ_w is re-written into a volumetric form over the advective bar as opposed to a

surface integral over the bar-coupled surface (for ease of implementation). Eq. (3.84) becomes:

$$\begin{aligned} R_P^i = & \underbrace{\int_{\mathrm{V}} \phi^i \hat{\boldsymbol{s}} \cdot (\rho \boldsymbol{v}_{bar} \cdot \boldsymbol{\nabla} \boldsymbol{v}_{bar} + \nabla P_{bar}) \, \mathrm{dV}}_{\mathrm{ADV}} \\ & + \underbrace{\int_{\mathrm{V}} \phi^i \left(\frac{W_p}{A} \tau_w - \hat{\boldsymbol{s}} \cdot \rho \boldsymbol{g} \right) \mathrm{dV}}_{\mathrm{SRC}} = \boldsymbol{0} \end{aligned} \tag{3.85}$$

where W_p is the wetted perimeter. Thus, any major losses are account for in the source term. Currently, the wall shear stress is modeled using the Darcy-Weisbach equation:

$$\tau_w = \frac{1}{8} f \rho \boldsymbol{v}_{bar} \cdot \boldsymbol{v}_{bar} \tag{3.86}$$

where f is a non-dimensional friction factor. This friction factor is defined in the Aria material block; for the bar momentum equation, the Haaland friction factor is typically used:

$$f = \begin{cases} \left[-1.8 \log_{10} \left(\left(\frac{r_\varepsilon}{3.7D} \right)^{1.11} + \frac{6.9}{Re} \right) \right]^{-2} & Re > 2000 \\ \frac{64}{Re} & 0 < Re \leq 2000 \end{cases} \tag{3.87}$$

where D is the hydraulic diameter and r_ε is the relative surface roughness. Eq. (3.87) is a fitting to the Moody chart, which is a graph of friction factors for varying Reynolds number Re and surface roughness r_ε. Some notes on defining this friction factor are reviewed in *Current Discrepancies with Advective Bars*.

Level Set

The level set method resolves evolving interfaces between multiple phases within a domain; this is done by tracking a smooth signed distance function Φ over the domain that indicates the closest distance to the interface at any point. An example of a Φ function is given for a 2D circular interface ($\Phi = \sqrt{x^2 + y^2} - 1$), is shown in Fig. 3.2

As seen in Fig. 3.2, values where $\Phi < 0$ indicate one phase, and $\Phi > 0$ indicates another; the interface separating the two phases is represented by the $\Phi = 0$ isocontour. Geometric parameters such as the interface normal $\boldsymbol{n}$ and curvature κ can be calculated directly from Φ since it is a smooth function:

$$\boldsymbol{n} = \frac{\nabla \Phi}{\|\nabla \Phi\|}, \quad \kappa = \nabla \cdot \boldsymbol{n} \tag{3.88}$$

An important property of Φ is that it remains a signed distance function; this ensures that the computations in (3.88) are accurate. This property is enforced by ensuring the norm of the gradient of Φ is equal to 1:

$$\|\nabla \Phi\| = 1 \tag{3.89}$$

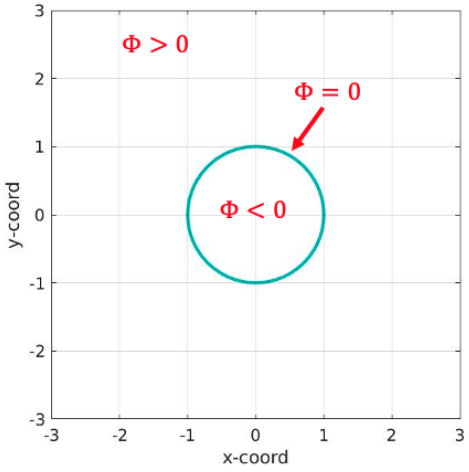


Fig. 3.2: Schematic of 2D circular interface using a signed distance function

The variable Φ is typically advected with the fluid velocity $\boldsymbol{v}$, which is obtained from the solution of (3.22):

$$\frac{\partial \Phi}{\partial t} + \boldsymbol{v} \cdot \nabla \Phi = 0 \tag{3.90}$$

In general, Eq. (3.90) does not satisfy the property posed by Eq. (3.89). An additional redistancing operation must be performed throughout the simulation, which will be discussed in *Level Set/CDFEM*. Aria allows the user to define generic sources S_V to model any sinks/production rates into the level set field, thus (3.90) is modified to:

$$\underbrace{\frac{\partial \Phi}{\partial t}}_{\text{MASS}} + \underbrace{\boldsymbol{v} \cdot \nabla \Phi}_{\text{ADV}} - \underbrace{S_V}_{\text{SRC}} = 0 \tag{3.91}$$

Notes on Solid Mechanics

Some of the standard references on solid mechanics include [4], [5], [6] and [7]. As is often the case, the mathematical notion used through-out these texts is different in many cases and this is often a source of confusion. Here, we'll lay out some basic definitions in our notation and, when possible, give the notation used in these other texts.

In what follows, $\boldsymbol{x}$ is the position vector of a material particle in the *deformed* or *current* spatial configuration and $\boldsymbol{X}$ is the position vector of a material particle in the *undeformed* or *initial* or *reference* configuration. The displacement vector $\boldsymbol{d}$ is the difference between these to states[1] viz. $\boldsymbol{x} = \boldsymbol{X} + \boldsymbol{d}$.

[1] $\boldsymbol{d}$ is denoted $\boldsymbol{u}$ in [4], [5], [6], and [7].

We will make extensive use of the gradients of these fields and so it is important to distinguish between gradients in the reference configuration and the current configuration. Gradients in the current configuration are denoted $\boldsymbol{\nabla}$ in Gibbs notation or $\partial\ /\partial x_i$ in index notation. Gradients in the reference configuration are denoted $\boldsymbol{\nabla}_X$ in Gibbs notation or $\partial\ /\partial X_i$ in index notation[2].

Next, we define the deformation gradient $\boldsymbol{F}$

$$\begin{aligned}\boldsymbol{F} &\equiv \boldsymbol{\nabla}_X \boldsymbol{x}^t \\ &= \boldsymbol{\nabla}_X \boldsymbol{X}^t + \boldsymbol{\nabla}_X \boldsymbol{d}^t \\ &= \boldsymbol{I} + \boldsymbol{\nabla}_X \boldsymbol{d}^t\end{aligned}$$

where the superscript ^{t} denotes the transpose operator[3]. The inverse deformation gradient[4], $\boldsymbol{F}^{-1}$, is also useful and can be computed as

$$\begin{aligned}\boldsymbol{F}^{-1} &\equiv \boldsymbol{\nabla} \boldsymbol{X}^t \\ &= \boldsymbol{\nabla} \boldsymbol{x}^t - \boldsymbol{\nabla} \boldsymbol{d}^t \\ &= \boldsymbol{I} - \boldsymbol{\nabla} \boldsymbol{d}^t.\end{aligned}$$

The determinants of $\boldsymbol{F}$ and $\boldsymbol{F}^{-1}$ are denoted J and J^{-1} respectively and are often used in transformations between different stress definitions[5].

It's worth noting at this point that in Aria both gradient operators, $\boldsymbol{\nabla}$ and $\boldsymbol{\nabla}_X$, are available as Expression objects as are $\boldsymbol{F}$, $\boldsymbol{F}^{-1}$, J and J^{-1}.

The Green or Green-Lagrange strain tensor $\boldsymbol{E}$ is defined[6] as

$$\begin{aligned}\boldsymbol{E} &\equiv \frac{1}{2}\left(\boldsymbol{F}^t \cdot \boldsymbol{F} - \boldsymbol{I}\right) \\ &= \frac{1}{2}\left(\boldsymbol{\nabla}_X \boldsymbol{d} + \boldsymbol{\nabla}_X \boldsymbol{d}^t + \boldsymbol{\nabla}_X \boldsymbol{d} \cdot \boldsymbol{\nabla}_X \boldsymbol{d}^t\right).\end{aligned}$$

The Green strain is a strain measure in the reference configuration and is suitable for large deformations and large rotations. The analogous Eulerian (or Almansi's) strain tensor $\boldsymbol{E}^*$ is defined[7] as

$$\begin{aligned}\boldsymbol{E}^* &\equiv \frac{1}{2}\left(\boldsymbol{I} - \boldsymbol{F}^{-t} \cdot \boldsymbol{F}^{-1}\right) \\ &= \frac{1}{2}\left(\boldsymbol{\nabla} \boldsymbol{d} + \boldsymbol{\nabla} \boldsymbol{d}^t - \boldsymbol{\nabla} \boldsymbol{d} \cdot \boldsymbol{\nabla} \boldsymbol{d}^t\right).\end{aligned}$$

The Eulerian strain is also a suitable strain measure for large deformations and rotations but is defined in the current configuration.

[2] In [7] $\boldsymbol{\nabla}_X$ is denoted $\boldsymbol{\nabla}_0$.

[3] In [5] this is called the *conjugate dyadic* and is denoted with a subscript c. In [4] the quantity $\boldsymbol{\nabla}_X \boldsymbol{x}^t$ is denoted $\boldsymbol{x}\ \overleftarrow{\boldsymbol{\nabla}}_X$ where the arrow over the gradient operator denotes the direction of the operation.

[4] In [5] $\boldsymbol{F}^{-1}$ is denoted $\boldsymbol{H}$.

[5] Sometimes J is expressed as a ratio of the densities between the reference and current configurations, $\rho_\circ/\rho$.

[6] In [5] $\boldsymbol{E}$ is denoted $\boldsymbol{L}_G$ and is called the *Lagrangian strain*.

[7] In [5] $\boldsymbol{E}^*$ is denoted $\boldsymbol{E}_A$.

The Cauchy stress, $\boldsymbol{\sigma}$, is a stress measure defined in the current configuration as

$$\boldsymbol{\sigma} = \lambda E_{kk} \boldsymbol{I} + 2\mu \boldsymbol{E}$$

where E_{kk} denotes the trace of $\boldsymbol{E}$ and λ and μ are the Lamé coefficients. This constitutive equation may also be augmented with some initial residual stress or a thermal stress

$$\boldsymbol{\sigma} = \lambda E_{kk} \boldsymbol{I} + 2\mu \boldsymbol{E} - \beta \left(T - T_{\circ}\right) \boldsymbol{I} + \boldsymbol{\sigma}_r.$$

Here β is related to the coefficient of thermal expansion, T is the temperature, $T_{\circ}$ is the reference temperature of the solid and $\boldsymbol{\sigma}_r$ is the residual stress.

For large deformations and large rotations Aria uses the second Piola-Kirchhoff stress which is defined in the reference configuration and is related to the Cauchy stress as

$$\boldsymbol{S} = J\boldsymbol{F}^{-1} \cdot \boldsymbol{\sigma} \cdot \boldsymbol{F}^{-t}.$$

The reverse transformation is readily given by

$$\boldsymbol{\sigma} = J^{-1}\boldsymbol{F} \cdot \boldsymbol{S} \cdot \boldsymbol{F}^{t}.$$

Mathematically, the Cauchy stress $\boldsymbol{\sigma}$ is most conveniently expressed in terms of the Lamé coefficients λ, μ and β. In practice, however, it is more common to measure and report a different but related set of parameters: the Young's modulus E, the Poisson's ratio ν and the coefficient of thermal expansion α. (Note, the shear modulus $G = \mu$.) The relationship between these two sets of parameters is

$$\begin{aligned} 2\mu &= \frac{E}{(1+\nu)} \\ \lambda &= \frac{\nu E}{(1+\nu)\,(1-2\nu)} &&= 2\mu \frac{\nu}{(1-2\nu)} \\ \beta &= \frac{\alpha E}{(1-2\nu)} &&= \alpha\,(3\lambda + 2\mu) \end{aligned}$$

In Aria, there are separate *Expression_Name*s for the Cauchy stress and the second Piola-Kirchhoff stress expressions. In the input files, users provide their choice of constitutive relations in the material model specification, e.g.,

```
Begin Aria Material The_Material
   Density     = Constant rho = 2.33e-15
   Mesh Lambda = Constant lambda = 52810.30445
   Mesh Two Mu = Constant two_mu = 134426.2295
   Mesh Stress = Nonlinear_Elastic Reference_Frame=Moving
   Mesh Stress = Residual Sx=-11 Sy=-11
   Mesh Stress = Isothermal T=800 T_ref=450
End
```

In this Example, there will be three contributions to the stress in the mesh stress: nonlinear elasticity, a planar residual stress and an isotropic linear thermal stress. Internally, Aria contains a separate expression for transforming these Cauchy stresses into Piola-Kirchhoff stresses, viz.

$$\boldsymbol{S} = J\boldsymbol{F}^{-1} \cdot \left(\sum_i \boldsymbol{\sigma}_i \right) \cdot \boldsymbol{F}^{-t}. \tag{3.92}$$

Note that second Piola-Kirchhoff stresses are not specified in the input file – only Cauchy stresses are. Aria automatically creates an Expression to compute the transform in equation (3.92).

3.3 Boundary Conditions

This section provides a high-level overview of the types of boundary conditions that can be applied in Aria. These boundary conditions are often denoted as Dirichlet, Neumann or Robin conditions. Dirichlet or essential boundary conditions for a solution field stem directly from the governing equations for a particular physics. In Aria, Neumann and Robin conditions are generally categorized as flux boundary conditions.

Dirichlet (Specified Value)

The Dirichlet boundary condition is probably the most straightforward boundary condition to apply. Since, by definition, it is not a function of the unknown solution field, its complexity is limited to being a known function of space and time. The specified field must be continuous on the surface of each subdomain, $\partial\Omega_i$. See the *command reference* for a complete list of ways to specify a dirichlet BCs.

Neumann (Specified Flux)

The flux boundary condition may be written (for the flux q_n of conserved quantity b) as

$$\boldsymbol{\nabla} b \cdot \hat{\mathbf{n}} \sim q_n = q_b\left(\mathbf{x}, t, b\right), \tag{3.93}$$

where $q_b(\mathbf{x}, t, b)$ is a given function. Note that an adiabatic boundary condition (default) defines a zero flux boundary condition:

$$q_n = 0$$

See the *command reference* for a complete list of ways to specify a flux BCs.

Fluxbp

Fluxbp provides a flux condition integrated by parts, with the primary use case in Aria for modeling Capillary forces in CDFEM simulations (see *CDFEM Approach to Modeling the Capillary Force* for more details). In this boundary condition, the flux condition is incorporated via integration by parts, which is beneficial in avoiding higher-order derivatives. For example, in multiphase flows involving capillary forces, using Fluxbp conditions can avoid higher-order derivatives associated with calculating the local curvature.

Disting

Aria's distinguishing condition (DC) feature is an essential ingredient in solving many coupled physics problems. A distinguishing condition is really just another equation specification except that it typically replaces a regular equation on a subset of the domain such as a surface.

For example, the temperature can be constrained to a specified wall temperature via a polynomial or string function to specify the following

$$T - T_w = 0 \tag{3.94}$$

where T_w is a reference wall temperature.

An additional feature of distinguishing conditions is that multiple DCs for a given degree of freedom on a given surface are added together. This additive feature allows users to build up their own conditions from primitive ones using string functions (see *String Functions*), where any zero-valued function can serve as a distinguishing condition.

An important thing to know about these conditions is that they are satisfied *weakly* as

$$\int_{\partial\Omega\setminus\Gamma_\phi} w\,(T - T_w)\; d\Gamma. \tag{3.95}$$

Consequently, the condition is only satisfied weakly and to within the tolerance of the nonlinear solver.

Rotated

Rotated boundary conditions are specialized forms of distinguishing conditions. For example, in solving fluids problems with a free surface where the mesh boundary moves with the material, e.g., an ALE simulation, a kinematic condition is used to tie the mesh to the fluid on the free boundary. In this case, the velocity normal to the free surface satisfies

$$\boldsymbol{n} \cdot \left(\boldsymbol{v} - \dot{\boldsymbol{d}}\right) - v_\circ = 0. \tag{3.96}$$

In this case, the matrix rows associated with the ordinal directions of the equation are rotated prior to applying the distinguishing conditions, such that the matrix rows now correspond to normal and

tangential directions instead of the original ordinate directions. For (3.96), the rows associated with the mesh degrees of freedom would be rotated after finite element assembly, but before applying the distinguishing condition, and (3.96) only contributes to a single matrix row which now corresponds to the normal direction. The rotation is then undone such that the full solution is still represented in the problem coordinate system, and physics associated with tangential directions were not affected.

Interface

Interface boundary conditions are used to provide additional contributions analogous to flux or disting boundary conditions on an interface between two materials (interface and interface_disting, respectively). Interface boundary conditions are applied to contiguous meshes. These contributions are equivalent to those in (3.93); however, the flux can now have a functional relationship on quantities on both sides of the interface. If the same degree of freedom is used on both sides of the interface, by construction the finite element volume assembly will account for the flux through the interface, so any contribution from the interface BC would be an additional contribution. However, there are situations where the volume contribution is not done on both sides of the interface, e.g., when both volume blocks are in separate regions or when the volume blocks are in different material phases. One example of this kind of usage is for Robin-style coupling in conjugate heat transfer problems.

Colloc Prefix

The colloc keyword works for distinguishing and rotated boundary conditions. Instead of applying the penalty function weakly, i.e., weighted by the finite element function and integrated over the surface, the boundary condition is applied nodally via a lumped approximation.

Warning: While colloc boundary conditions should effectively be the same as lumped boundary conditions, some inconsistencies have been noted in how colloc boundary conditions are implemented. Namely, `COLLOC_ROTATED` seems to mirror a lumped approximation for a rotated boundary condition, while `INERFACE_COLLOC_DISTING` seems to use a simple (unweighted) nodal contribution

Lumped Prefix

The lumped prefix is applicable to flux and disting style boundary conditions. For `LUMPED_FLUX`, instead of using a consistent finite element integration, the flux contributions are lumped at the nodes similarly to a lumped mass approximation. As mentioned in the previous section, `LUMPED_DISTING` provides a lumped version of the distinguishing condition, except when in conjunction with a `INTERFACE_COLLOC_DISTING`. More details can be found in *Mass Lumping*.

3.4 Sources

This section discusses the volumetric source terms, as defined in *Equations Aria Solves* for each equation Aria solves. This term may vary in space and time, and is defined piecewise over each subdomain Ω_i. Physically, this term corresponds to the production rate of the conserved quantity b.

Note that this section is not a comprehensive list of all sources available in Aria, but is rather a subset of representative sources that are commonly used. Details on what a specific source applies can be found in its *command summary*.

3.5 Interface Conditions

In Fig. 3.1 an example of an interface between two subdomains of a problem was demonstrated. When the properties of these two subdomains are different, *Contact* or an *interface boundary condition* can be used to describe the abrupt change in conditions. At the interface between two subdomains Ω_i and Ω_j, we assume a zero jump in the flux of a quantity in the direction normal to the interface,

$$[[q_n]] = 0 \tag{3.97}$$

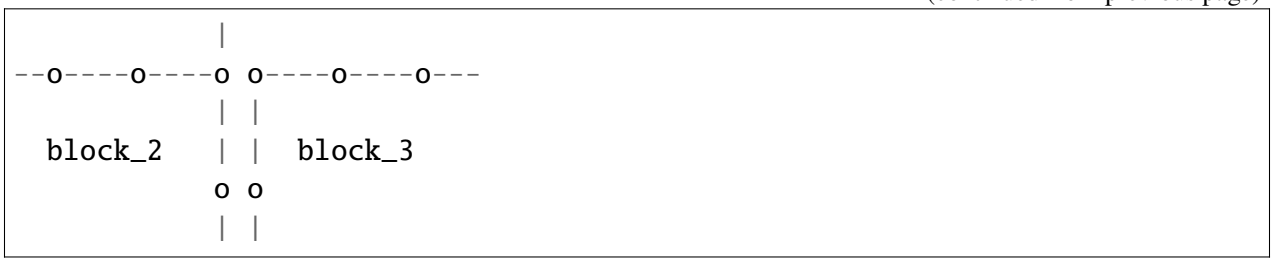
The follow section provides some examples of treatment of the conserved quantity at the interface, following this assumption.

It is worth noting that interface conditions can be applied in one of two ways, depending on the interface. If the mesh is contiguous at an interface (i.e. nodes are shared by the blocks connected to the interface), an *interface boundary condition* is sufficient. However if the nodes are not shared (as is demonstrated in Fig. 3.8), a contact definition must be made at the interface (see *Contact*). For example in the sketch below, one would need contact for a jump between `block_2` and `block_3`, but only an interface condition for a jump between `block_1` and `block_2`

```
            |
            o
  block_1   |
```

(continues on next page)

(continued from previous page)



Note that this section is not a comprehensive list of all contact enforcement conditions available in Aria, but is rather a subset of representative contact enforcement conditions that are commonly used. Details on what a specific boundary condition applies can be found in its *command summary*. Similarly, refer to *Contact* for information on how to setup a contact definition.

Perfect or Tied Contact

The first and most common condition is to enforce continuity of a conserved quantity b at the interface,

$$b\left|_{\partial\Omega_i} - b\right|_{\partial\Omega_j} = 0, \tag{3.98}$$

where the notation $b\,|_{\partial\Omega_i}$ indicates that the temperature is to be evaluated on the surface $\partial\Omega_i$, which is associated with the subdomain Ω_i.

Contact Resistance

The second condition allows for a model of imperfect contact between two surfaces, which can take account of surface roughness. The physical meaning of this contact resistance depends based on the equation it is applied to. In general, though, this model accounts for an imperfect mating of the two contact surfaces causing effective flux of the conserved quantity to occur only on a subset of the contacting surface.

We treat this contact by setting the "gap" flux across the interface proportional to the drop in the conserved quantity,

$$q_n|_{\partial\Omega_i} - q_n|_{\partial\Omega_j} = h_c\left(b\left|_{\partial\Omega_i} - b\right|_{\partial\Omega_j}\right), \tag{3.99}$$

where the constant h_c is the contact conductance, (analogous to a heat transfer coefficient, for a thermal problem). The value of contact conductance is dependent upon the following[1] :

- values of b of the two materials at the contact surface;
- the materials in contact;

[1] More on the theory of contact conductance, and the values of conductances for typical surface finishes and moderate contact pressures can be found in [15, 16, 17]

- surface finish and cleanliness;
- pressure at which the surfaces are forced together;
- the substance, or lack of it, in the interstitial spaces.

3.6 Radiation

Surface Radiation

A surface may exchange energy with its surroundings through thermal radiation. Any incident surface radiation will be either transmitted, reflected or absorbed. Letting τ, ρ and α represent the fractions of the incident flux in each category then

$$1 = \tau + \rho + \alpha$$

and for no transmission

$$1 = \rho + \alpha \text{ or } \rho = 1 - \alpha \ .$$

Using the Kirchhoff identity

$$\alpha = \epsilon$$

then the reflectance is

$$\rho = 1 - \epsilon \tag{3.100}$$

where ϵ is the emissivity.

In order to understand the radiative energy balance at a surface one considers the rate at which energy streams away from the surface, the radiosity, defined as

$$J = \epsilon E_b + \rho G \tag{3.101}$$

where E_b is the blackbody emissive power and G is the irradiation. Substituting for the reflectance (3.100) then

$$J = \epsilon E_b + (1 - \epsilon) G \ . \tag{3.102}$$

The surface flux q is the difference between the energy that radiates away and the incident energy

$$q = J - G \tag{3.103}$$

and substitution for the radiosity we find that

$$q = \epsilon(\sigma T^4 - G) \ .$$

When G is derived from an external temperature interaction this boundary condition is often called far-field radiation, since it usually models the radiative transfer of energy between a surface and the external environment. However, the boundary condition is found to have more general utility when one considers its role in modeling radiative transfer between two surfaces, 1 & 2, as shown in Fig. 3.3, where A_1 is analogous to $\partial\Omega_i$.

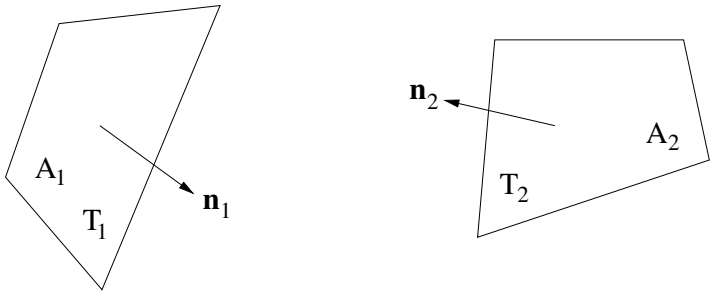


Fig. 3.3: Surface Radiative Exchange

For the case in which the temperature T_2 is known and independent of temperature T_1 then using the emissive power σT^4 the normal flux per unit area across A_1 may be written as

$$q = \sigma\epsilon F\left(T_1^4 - T_2^4\right), \tag{3.104}$$

where σ denotes the Stefan-Boltzmann constant, ϵ is the emissivity of the surface A_1 and F is the form factor. We remark that (3.104) is a nonlinear boundary condition, since the unknown temperature, T_1 is raised to the fourth power and furthermore the emissivity may be a function of temperature. It is important to note that the form factor F may differ from the more familiar *view factor* F_{12} encountered in enclosure radiation problems. The question often asked is how does one determine the appropriate value of form factor F.

The form factor F can be best described using a network analogy of radiative transfer between two surfaces as shown below.

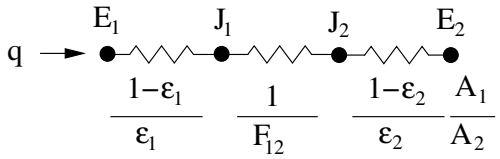


Fig. 3.4: Radiative Transfer Circuit Model

Using the emitted energy E and radiosity J the network the heat flux can be written in terms of a thermal resistance R as

$$q = \sigma\frac{T_1^4 - T_2^4}{R} \tag{3.105}$$

and from the network model

$$R = \frac{1-\epsilon_1}{\epsilon_1} + \frac{1}{F_{12}} + \frac{1-\epsilon_2}{\epsilon_2}\frac{A_1}{A_2} = \left(\frac{1}{\epsilon_1} - 1\right) + \frac{1}{F_{12}} + \left(\frac{1}{\epsilon_2} - 1\right)\frac{A_1}{A_2} \quad .$$

For $A_2 \gg A_1$ the third term of R can be neglected and $F_{12} = 1$. Comparing expressions (3.104) and (3.105) then

$$R = \frac{1}{\epsilon_1} \text{ and } F = 1 \quad .$$

so estimation of F_{12} is not required.

For $\epsilon_2 = 1$ (black receiving surface) but A_2 not $\gg A_1$ and once again the third term of R can be neglected so that

$$R = \frac{1-\epsilon_1}{\epsilon_1} + \frac{1}{F_{12}} = \frac{(1-\epsilon_1)F_{12} + \epsilon_1}{\epsilon_1 F_{12}} \text{ and } F = \frac{F_{12}}{(1-\epsilon_1)F_{12} + \epsilon_1} \quad . \tag{3.106}$$

If both surfaces are black $\epsilon_1 = \epsilon_2 = 1$ then from the previous expression (3.106) we find that $F = F_{12}$.

During a simulation the surface heat flux is integrated over the spatial discretization of surface A_1. Here we note that defining the flux on a per unit area basis enables us to apply the radiative flux (3.104) consistently with F evaluated for the entire surface A_1 even when the surface is discretized.

Enclosure Radiation

When energy radiates from one portion of a surface to another, and the intermediate medium is transparent (i.e., it does not absorb any energy), then enclosure radiation may be used to model the heat flux on the surface. Using the net radiation method [18], the normal flux at a particular location on the surface may be written as the difference between the emitted radiative heat flux leaving the surface, and the absorbed incident radiative flux due to the rest of the enclosure, namely

$$q_n = \sigma \epsilon T^4 - \alpha G, \tag{3.107}$$

where α denotes the absorptivity of the surface, and G represents the surface irradiation. Under the additional assumption that the emissivity, absorptivity, and reflectivity are independent of direction and wavelength, we may write

$$\epsilon = \alpha = 1 - \rho, \tag{3.108}$$

where ρ is the reflectivity.

Without loss of generality, we can regard the enclosure, $\Gamma_{\mathcal{E}}$, as a union of E surfaces,

$$\Gamma_{\mathcal{E}} = \Gamma_1 \cup \Gamma_2, \ldots \Gamma_{E-1} \cup \Gamma_E$$

This situation is illustrated in Fig. 3.6, where the radiosity for surface i in the enclosure is defined to be

$$J_i = \sigma \epsilon_i T_i^4 + \rho_i G_i, \tag{3.109}$$

where T_i is the area averaged constant temperature on facet Γ_i computed via

$$T_i = \frac{\sum_{j=1}^{N} T_j \int_{\Gamma_i} \psi_j \, d\Gamma}{\int_{\Gamma_i} d\Gamma} \tag{3.110}$$

The surface irradiation for surface i is determined by the radiosity of all the other surfaces in the enclosure through the relation

$$G_i = \sum_{j=1}^{E} F_{ij} J_j, \tag{3.111}$$

where F_{ij} denotes the geometric viewfactor of surface i with respect to surface j. The viewfactor may be considered as the fraction of energy that leaves surface j and arrives at surface i. For a given pair of surfaces shown in Fig. 3.5, the viewfactor is defined as the following integral

$$F_{ij} = \frac{1}{A_i} \int_{A_i} \int_{A_j} \frac{cos\theta_i \cos\theta_j}{\pi r^2} \; \delta_{ij} dA_i dA_j \quad . \tag{3.112}$$

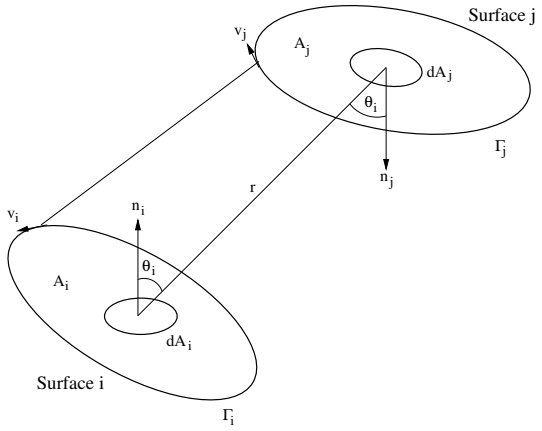


Fig. 3.5: Viewfactor Configuration.

Viewfactors are computed using the Chaparral library [19]. Determination of the viewfactors is a compute intensive endeavor. As such, extraneous calculations are eliminated based upon the geometry. One example of this would be excluding this calculation for surfaces which are not visible to each other. Moreover, from a geometric view of enclosure surfaces, Fig. 3.5, one can conclude that legitimate interactions between surfaces are those for which n_i and n_j are opposed. Thus an important feature of the enclosure model is the notion of inward facing normals. This convention effectively defines the interaction between the enclosure subfacets.

Note: For closed surfaces (watertight enclosures), for each facet i the row-sum over all surface N facets should equal one, i.e., $\sum_{j=1}^{N} F_{ij} = 1.0$.

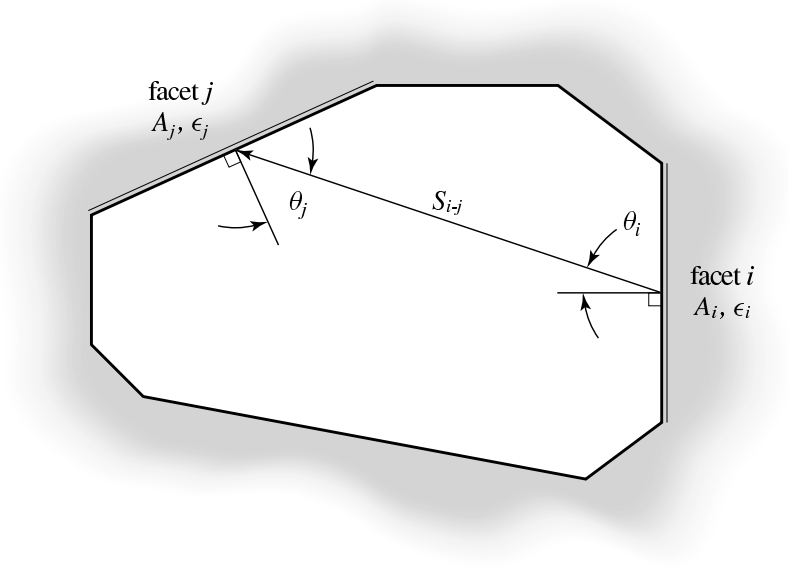


Fig. 3.6: Two arbitrary facets radiating energy to one another in a *radiation enclosure*. The energy exchanged depends on: the shape, orientation, distance, area A_i, A_j, temperatures T_i, T_j, and radiative properties of the facets ϵ_i, ϵ_j.

Upon substitution of equations (3.111) and (3.108) into (3.109), the radiosity may be written as

$$J_i - (1 - \epsilon_i) \sum_{j=1}^{N} F_{ij} J_j = \sigma \epsilon_i T_i^4, \tag{3.113}$$

Finally, the first J_i term in (3.113) may be moved inside the summation to yield

$$\sum_{j=1}^{N} \left[\delta_{ij} - (1 - \epsilon_i) F_{ij} \right] J_j = \sigma \epsilon_i T_i^4, \tag{3.114}$$

where

$$\delta_{ij} = \begin{cases} 1 & \text{if } i = j \\ 0 & \text{if } i \neq j \end{cases}$$

(3.114) is a system of equations that can be solved for the unknown radiosity at each face. Finally, we may rewrite (3.107) to express the radiative heat flux on surface i as

$$q_n = \sigma \epsilon_i T_i^4 - \epsilon_i G_i, \tag{3.115}$$

where G_i is given by (3.111) and depends on the unknown radiosity. This heat flux is applied as a boundary condition to the appropriate energy equations, thereby fully coupling the enclosure radiation problem with the volume energy equations.

Instead of solving the monolithic equation system by forming the jacobian for the combined coupled system of equation systems, a segregated approach is used. At each nonlinear iteration,

the radiosity equation is solved separately with the right hand side computed from the average facet temperature (3.110) using the current nonlinear temperature solution. Subsequently, the irradiation is computed from the radiosity in order to compute the heat flux contribution given by (3.115). A Newton step is then performed where the jacobian contribution of the radiative flux is computed as

$$\frac{\partial R_i}{\partial T_j} = \int \phi_i \frac{\partial q_n}{\partial T_j} d\Gamma, \tag{3.116}$$

where R_i is the i^{th} nonlinear temperature residual, ϕ_i the test function and where

$$\frac{\partial q_n}{\partial T_j} = \frac{\partial q_n}{\partial T^e} \frac{\partial T^e}{\partial T_j}. \tag{3.117}$$

Here, the facet temperature T_i given by (3.110) has been relabeled to T^e to avoid confusion with the nodal finite element temperature solution. The sensitivity of the radiative heat flux with respect to the facet temperature is approximated as

$$\frac{\partial q_n}{\partial T^e} = 4\sigma\epsilon^e \left[T^e\right]^3 \tag{3.118}$$

while the sensitivity of the facet temperature with respect to the nodal temperature dof is

$$P_j^e = \frac{\partial T^e}{\partial T_j} = \frac{\int_{\Gamma^e} \phi_j d\Gamma}{A^e}, \tag{3.119}$$

where A^e is the area of a facet. In terms of the element contribution, this can be compactly written as

$$\frac{\partial R_i^e}{\partial T_j} = 4\sigma\epsilon^e \left[T^e\right]^3 A^e P_i^e P_j^e \tag{3.120}$$

Note: The average facet temperature is $T_e = \sum_j P_j^e T_j$. For computing the cube $[T^e]^3$, experience shows projecting the raised power i.e., $[T^e]^3 = \sum_j P_j^e \left[T_j\right]^3$ leads to less spurious oscillations.

Banded Wavelength Enclosure Radiation

The enclosure radiation model previously presented considers a net response over all wavelengths assuming that all surfaces behave as grey bodies. Here the surface response with regard to different wavelengths is implicitly included in the model by using temperature dependent emissivities and allowing the surfaces to emit as grey bodies. In this case the surface flux previously mentioned (3.107) can be alternatively expressed as

$$q_n = \int_0^\infty q(\lambda) \ d\lambda. \tag{3.121}$$

In many applications engineering materials respond differently to different portions of the thermal energy spectrum. Thus modeling the thermal radiation response of these materials using the entire blackbody spectrum results in poor characterization of the enclosure response. For this purpose more specialized approaches for radiation modeling of wavelength dependent surfaces have been developed using continuous representations of emissivity as a function of wavelength and temperature while integrating over the wavelength spectrum. For numerical modeling the simplest approach involves discretization of the wavelength spectrum into a few representative bands N_b and integrating over each of the k bands. Using this approach the methods previously described follow directly except that the emissivity and emissive power now have independent representations in a wavelength band, each of which contributes some surface flux Δq_k thus

$$q_n = \sum_{k=1}^{N_b} \Delta q_{k\lambda}. \tag{3.122}$$

Recalling that flux can also be expressed in terms of radiosity then similarly the radiosities are obtained by solving a system of equations

$$\sum_{k=1}^{N_b} \sum_{j=1}^{N} \left[\delta_{(ik)j} - (1 - \epsilon_{(ik)}) F_{ij}\right] \Delta J_{jk} \Delta\lambda_k = \sum_{k=1}^{N_b} \mathrm{F}_{\Delta\lambda_k} \sigma \epsilon_{(ik)} \left[T_i\right]^4 \Delta\lambda_k \tag{3.123}$$

where ΔJ_{jk} is the incremental radiosity and $\mathrm{F}_{\Delta\lambda_k}$ is the blackbody fraction for band k. Finally we note that given a fixed facet temperature field this solution can be carried out independently for each wavelength band and the radiosities ΔJ_{jk} can be accumulated to obtain the net facet radiosity J_j and net flux q_n. Thus the banded wavelength model very much resembles the more simplified *enclosure radiation* model. From a simulation point of view, the major difference being that the modeler must supply information describing the band discretization in addition to emissivity models for each band.

Construction of the banded wavelength model begins by first prescribing the emissivity variation for each surface of the enclosure as a function of wavelength λ. The overall band discretization is then obtained by collective consideration of the wavelengths at which the emissivity changes on any of the given surfaces. Thus the number of bands for the enclosure will likely be more than those for any single surface and is demonstrated in Fig. 3.7 below.

Note: Each modeled band must have an emissivity model supplied for each surface, even when its emissivity is not changing.

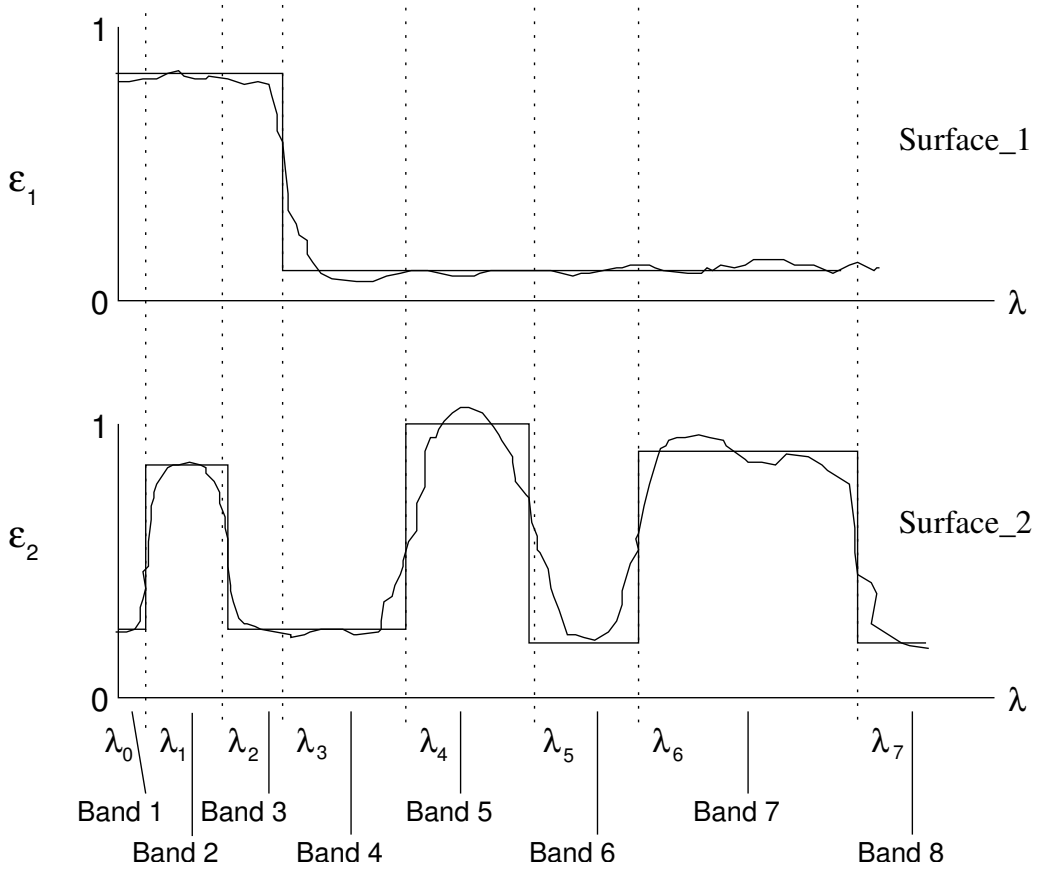


Fig. 3.7: Emissivity bands for two-surface problem.

3.7 Contact

The term "contact" refers to a situation where a simulation needs to be performed that spans a non-contiguously meshed interface, i.e. a nonconformal interface. One example of a nonconformal interface is one that has adjacent blocks and there appears to be a single node at a certain location in space on the interfaces between blocks, but in fact there are distinct nodes that have identical, or nearly identical coordinates. In other words, the mesh nodes have not been merged or equivalenced. More generally, a nonconformal interface does not require that mesh lines match at the interface between block boundaries.

There are two modeling situations that lead to the use of nonconformal interfaces in analyses. The first occurs when compatibility conditions are enforced at the interface to require that the conserved quantity be continuous across the interface. This situation arises as a convenient way to generate the mesh for particularly complex geometries since both parts can be meshed independently. The second modeling situation that occurs is not a matter of convenience, but of physics. It may be that two parts are known to fit imperfectly, so that there is a gap between the two parts that has a certain known resistance. In this case, the enforcement is known as gap conductance (or resistance) and enforces a discontinuous solution across the gap by modeling the flux that is "lost" across the gap. This situation is similar to *Interface Conditions* when the discontinuous properties exists at an interface that is in a contiguous portion of the mesh.

The contact algorithm is implemented as a Discontinuous Galerkin (DG)-like method that most closely follows the "Generalized Algorithm" discussed in [20] and references therein, and users are encouraged to read that report for more details regarding the nuances associated with the various types of contact enforcement.

Contact Interface

Contact is often enforced across an interface with a noncontiguous mesh, as seen in Fig. 3.8. The surfaces between two subdomains may be meshed independently, so that there may be a physical gap between them, or the nodes on the two surfaces may not align. Moreover, in parallel computations, these surfaces will in general have different processor decompositions so that all of the required information may not be available locally on a given processor. This nonlocality of information complicates the implementation, because this information must be constructed in a consistent way on all processors. Aria uses the `stk::search` and `Dash` packages to detect when facets that lie on separate surfaces are close enough to be in contact in a parallel consistent manner.

Even in cases of collocated nodes, the DG-nature of the method means that the conserved quantity is doubly defined along the contact interface, which is shown as Γ_{ij} in Fig. 3.9. The preferred contact enforcement strategy for the case with no contact resistance, i.e., physically a continuous conserved quantity field, is `TIED_CONTACT`, which is derived similarly to Discontinuous Galerkin (DG) and Interior Penalty (IP) methods [20, 21]. In practice, the continuity in the conserved quantity is only enforced weakly; however, the jump is convergent with mesh refinement. `TIED_CONTACT` is useful for analyzing systems whose geometry is so complex that a contiguous

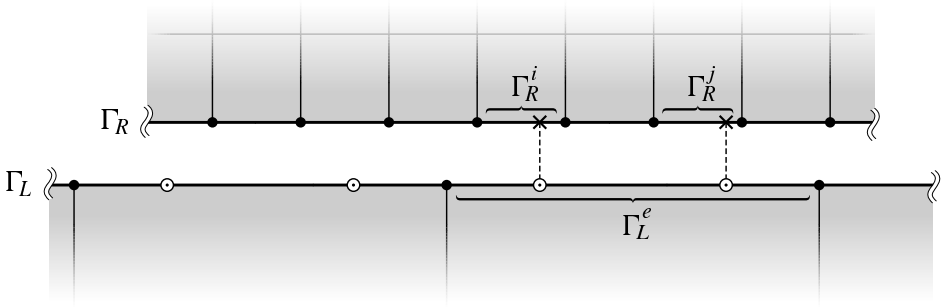


Fig. 3.8: In a Discontinuous Galerkin-based contact enforcement strategy, the elements involved on the right side Γ_R, are those whose sides intersect the quadrature points of the element on the left side Γ_L. In general, the sides of elements on the right side, Γ_R^i, and Γ_R^j, may not be contiguous.

mesh is difficult or impossible to obtain. Also available as an enforcement strategy is `GAP_CONDUCTANCE`, which allows a discontinuous conserved quantity across the interface, and specifies a flux according to a contact resistance. Contact resistance is generally used when the system being analyzed consists of at least two parts that touch, and it is important to model their imperfect fit with a finite resistance.

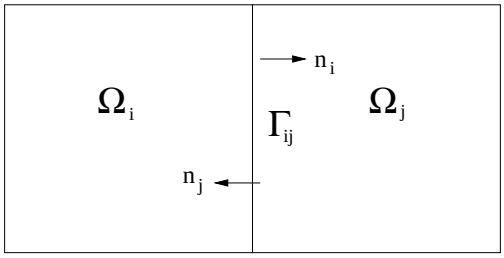


Fig. 3.9: Contact interface between subdomains Ω_i and Ω_j

Contact Enforcement

In Aria `TIED_CONTACT` and `GAP_CONDUCTANCE` are implemented in a variational form. Consider a 2D or 3D domain Ω that consists of a set of subdomains Ω_i with interfaces Γ_{ij} at the intersection of each pair Ω_i and Ω_j, $i \neq j$ where $n_{i,j}$ denote the unit outward normal vector on each subdomain boundary $\partial\Omega_{i,j}$, as shown in Fig. 3.9. On each subdomain Ω_i, we pose a standard scalar conduction problem of the form

$$\nabla \cdot \boldsymbol{q} = f, \quad x \in \Omega_i, \tag{3.124}$$

where $\boldsymbol{q}$ is the heat flux, and f is a volumetric source term, along with appropriate boundary conditions on $\partial\Omega_i/\Gamma$. The flux is defined as $\boldsymbol{q} \equiv -k\boldsymbol{\nabla} b$, where b is the scalar conserved quantity, and k is the conductivity. The notion of average and jump notations are useful in describing this method. The average notation is defined as

$$\{b\} = \frac{1}{2}(b_i + b_j)$$

for both scalar and vector quantities and the jump operator is defined as

$$[b] \equiv b_i \boldsymbol{n}_i + b_j \boldsymbol{n}_j$$

and

$$[\boldsymbol{q}] \equiv \boldsymbol{q}_i \cdot \boldsymbol{n}_i + \boldsymbol{q}_j \cdot \boldsymbol{n}_j$$

for scalar and vector variables, respectively.

To write (3.124) in a variational statement, let v be suitable a test function that is continuous within each subdomain and possibly discontinuous across Γ. We first multiply the differential energy (3.124) in each subdomain Ω_i by a suitable test function v and integrate by parts to get the weak form

$$\int_{\Omega_i} \boldsymbol{\nabla} v \cdot \boldsymbol{q} \, d\Omega + \int_{\Omega_i} v f \, d\Omega - \int_{\partial\Omega_i} v \left(\boldsymbol{q}_i \cdot \boldsymbol{n}_i \right) \, dS = 0$$

Assuming for simplicity Dirichlet boundary conditions for T have been specified on $\partial\Omega_i \backslash \Gamma$, we can assume that $v = 0$ on $\partial\Omega_i \backslash \Gamma$. Summing over domains gives

$$\int_{\Omega_i} \boldsymbol{\nabla} v \cdot \boldsymbol{q} \, d\Omega + \int_{\Omega_i} v f \, d\Omega - \sum_K \int_{\partial\Gamma_{ij}} v \left(\hat{\boldsymbol{q}}_k \cdot \boldsymbol{n}_k \right) \, dS = 0 \tag{3.125}$$

where $\hat{\boldsymbol{q}}$ is an arbitrary numerical flux to be chosen, and the summation is over all surface elements along the interface Γ_{ij}. Using the jump operators and noting that $[\hat{\boldsymbol{q}}] = 0$ is necessary for conservation, (3.125) can be written compactly as

$$\int_{\Omega} \boldsymbol{\nabla} v \cdot \boldsymbol{q} \, d\Omega_i + \int_{\Omega_i} v f \, d\Omega - \int_{\partial\Gamma_{ij}} [v] \cdot \{\hat{\boldsymbol{q}}\} \, dS = 0 \tag{3.126}$$

As discussed in [21], there are many valid choices for closing the numerical flux. One such method is the Interior Penalty method, which is the method listed as the "Generalized Algorithm" in [20]. In this case the numerical flux is chosen as $\hat{\boldsymbol{q}} \equiv -\alpha_h k \boldsymbol{\nabla} b + \beta_h [b]$, and this choice can be shown to be both consistent and conservative in the weak form of the problem. The resultant weak form of the problem becomes

$$\begin{aligned} &\int_{\Omega} \boldsymbol{\nabla} v \cdot k \boldsymbol{\nabla} b \, d\Omega_i - \int_{\Omega_i} v f \, d\Omega \\ &- \int_{\partial\Gamma_{ij}} [v] \cdot \{\alpha_h k \boldsymbol{\nabla} b\} \, dS + \int_{\partial\Gamma_{ij}} \beta_h [v] \cdot [b] \, dS = 0 \end{aligned} \tag{3.127}$$

For the `TIED_CONTACT` case (an Interior Penalty Method), coefficient $\alpha_h \equiv 1$ and β_h is defined as

$$\beta_h \equiv C \overline{k h^{-1}},$$

where the overline denotes an average from both sides of the interface Γ_{ij}, C is an arbitrary positive number, nominally $1/2$, and h is a local mesh length scale.

For the GAP_CONDUCTANCE case, α_h and β_h are chosen as in [20], which is repeated here for convenience as

$$(\alpha_h, \beta_h) \equiv \begin{cases} \left(1 - \frac{1}{H}, C\overline{kh^{-1}}\right) & H > 1, \\ \left(0, R^{-1}\right) & H \leq 1, \end{cases}$$

where

$$H = \frac{R^{-1}}{C\overline{kh^{-1}}}.$$

As before, the overline notation is used to denote the average from both sides of the interface.

While the above formulation is consistent, i.e. $\hat{\boldsymbol{q}}(b) = \boldsymbol{q}(b)|_\Gamma$, and conservative, i.e., $\hat{\boldsymbol{q}}$ is single valued along Γ, the discrete form of the integrals is only conservative in the case of collocated nodes, which is generally not the case in practical usage. However, [20] has shown that the error norms do decrease according to the order of the underlying finite-element scheme, as expected.

3.8 Discretization

In this chapter, we discretize the governing equations and their boundary conditions in space and time[1]. There are several approaches that can be used to accomplish this. Aria currently first semi-discretizes in space using the Galerkin method, and then discretizes in time using finite differences. Alternative approaches include first semi-discretizing in time using finite differences, then discretizing in space. Another possibility would be to discretize in space and time simultaneously using a space-time finite element formulation. For clarity, we will be using the stationary and transient forms of the *general conservation equation*.

In *Weak Statement of the Stationary Problem*, we use the method of weighted residuals[2] to obtain a weak statement for the general conservation equation, given by *general conservation equation*. We then use this weak statement to form the associated Galerkin approximations in *Galerkin Approximation for the Stationary Problem*; in *Weak Statement of the Transient Problem*, we form a similar weak statement for the transient problem; in *Galerkin Approximation for the Transient Problem*, we discretize the resulting weak statement in space using Galerkin's method and use finite differences to discretize in time. We continue the analysis in *Finite Element Approximation*, where we introduce the finite element approximations. The development in this chapter closely follows that of Becker, et al. [22]. The interested reader should consult this reference for a more detailed presentation.

[1] The development in this chapter closely follows that in *Finite Elements: An Introduction*, by Becker, et al. [22].

[2] This classical method is treated in depth in *The Method of Weighted Residuals and Variational Principles*, by Finlayson [24].

Weak Statement of the Stationary Problem

We begin with the stationary form of the *general conservation equation*, which we rewrite as

$$\boldsymbol{\nabla} \cdot (\phi \boldsymbol{v}) = -\boldsymbol{\nabla} \cdot \boldsymbol{f} + \phi_{\mathrm{V}} \tag{3.128}$$

and is therefore defined over each subdomain Ω_ι. In fact, because of the possibility of discontinuities in the material properties at the interface between two subdomains, the second derivatives of ϕ may not exist on such interfaces. This means that it is not possible to simply integrate (3.128) over the entire domain Ω. In fact, the lack of well-defined second derivatives is precisely the reason why this equation was written in each subdomain in the problem statement given by *the domain definition* in the first place. Accordingly, we begin by multiplying (3.128) by an arbitrary, admissible test function, w, and integrating the result over a single subdomain Ω_ι. This integral may be written as

$$\int_{\Omega_\iota} w \left[\boldsymbol{v} \boldsymbol{\nabla} \cdot \phi + \phi \boldsymbol{\nabla} \cdot \boldsymbol{v} + \boldsymbol{\nabla} \cdot \boldsymbol{f} - \phi_{\mathrm{V}} \right] \, dV = 0 \tag{3.129}$$

Next, in order to integrate the second term in (3.129) by parts, we introduce the following identity:

$$\int_{\Omega_\iota} \boldsymbol{\nabla} \cdot (w \boldsymbol{f}) \, d\Omega = \int_{\Omega_\iota} \boldsymbol{\nabla} w \cdot \boldsymbol{f} \, d\Omega + \int_{\Omega_\iota} w \boldsymbol{\nabla} \cdot f \, d\Omega \tag{3.130}$$

Upon substitution of (3.130) into (3.129)), we obtain

$$\int_{\Omega_\iota} (w \boldsymbol{v} \cdot \boldsymbol{\nabla} \phi - \boldsymbol{\nabla} w \cdot \boldsymbol{f} - w \phi_{\mathrm{V}}) \, d\Omega + \int_{\Omega_\iota} \boldsymbol{\nabla} \cdot (w \boldsymbol{f}) \, d\Omega = 0 \tag{3.131}$$

where the assumption of a solenoidal velocity field drops terms containing $\boldsymbol{\nabla} \cdot \boldsymbol{v}$. Next, we introduce the Gauss Divergence Theorem, which converts a volume integral to a surface integral, and may be written as

$$\int_{\Omega_\iota} \boldsymbol{\nabla} \cdot (w \boldsymbol{f}) \, d\Omega = \int_{\partial \Omega_\iota} w \boldsymbol{f} \cdot \boldsymbol{n} \, d\Gamma \tag{3.132}$$

Upon substitution of (3.132) into (3.131) and rearranging terms, we obtain

$$\int_{\Omega_\iota} (w \boldsymbol{v} \cdot \boldsymbol{\nabla} \phi - \boldsymbol{\nabla} w \cdot \boldsymbol{f} - w \phi_{\mathrm{V}}) \, d\Omega + \int_{\partial \Omega_\iota} w \boldsymbol{f} \cdot \boldsymbol{n} \, d\Gamma = 0. \tag{3.133}$$

The next step in obtaining the weak statement is to sum the contributions from (3.133) over each subdomain Ω_ι to obtain the integral over the entire domain, Ω. This is now valid, because no second derivatives appear in (3.133). Hence, we may write

$$\sum_{\iota=1}^{N} \int_{\Omega_\iota} (w \boldsymbol{v} \cdot \boldsymbol{\nabla} \phi - \boldsymbol{\nabla} w \cdot \boldsymbol{f} - w \phi_{\mathrm{V}}) \, d\Omega = - \sum_{\iota=1}^{N} \int_{\partial \Omega_\iota} w f_n \, d\Gamma, \tag{3.134}$$

where $f_n \equiv \boldsymbol{f} \cdot \boldsymbol{n}$.

The sum of the volume integrals on the left-hand side of (3.134) may be readily combined into a single integral, namely

$$\begin{aligned} \sum_{\iota=1}^{N} \int_{\Omega_\iota} (w\boldsymbol{v} \cdot \boldsymbol{\nabla}\phi - \boldsymbol{\nabla} w \cdot \boldsymbol{f} - w\phi_{\mathrm{V}}) \; d\Omega \\ = \int_{\Omega} (w\boldsymbol{v} \cdot \boldsymbol{\nabla}\phi - \boldsymbol{\nabla} w \cdot \boldsymbol{f} - w\phi_{\mathrm{V}}) \; d\Omega \end{aligned} \tag{3.135}$$

The sum of the surface integrals on the right-hand side of (3.134) requires more care. Consider that the surface integral associated with each subdomain Ω_ι consists of two parts. The first part is over the portion of $\partial\Omega_\iota$ which intersects the exterior surface $\partial\Omega$, which we denote $\partial\Omega_\iota \cap \partial\Omega$. The second part is what is left over, and consists of internal surfaces that divide one subdomain from another, which we denote $\partial\Omega_\iota \setminus \partial\Omega$. The sum of surface integrals on the right hand side of (3.134) is therefore

$$\sum_{\iota=1}^{N} \int_{\partial\Omega_\iota} w f_n \, d\Gamma = \sum_{\iota=1}^{N} \int_{\partial\Omega_\iota \cap \partial\Omega} w f_n \, d\Gamma + \sum_{\iota=1}^{N} \int_{\partial\Omega_\iota \setminus \partial\Omega} w f_n \, d\Gamma \tag{3.136}$$

Now, the sum of the integrals over the exterior surfaces $\partial\Omega_\iota \cap \partial\Omega$ is simply the integral over the entire boundary surface $\partial\Omega$. The second sum in (3.136) reduces to the sum of the integrals of the flux jumps $[\![f_n]\!]$ over each interior interface [22]. Since the problem statement specifies that these jumps are zero, (3.136) reduces to

$$\sum_{\iota=1}^{N} \int_{\partial\Omega_\iota} w f_n \, d\Gamma = \int_{\partial\Omega} w f_n \, d\Gamma \tag{3.137}$$

Upon substitution of (3.137) and (3.135) into (3.134), we have

$$\int_{\Omega} (w\boldsymbol{v} \cdot \boldsymbol{\nabla}\phi - \boldsymbol{\nabla} w \cdot \boldsymbol{f}) \; d\Omega = -\int_{\partial\Omega} w f_n \, d\Gamma + \int_{\Omega} w\phi_{\mathrm{V}} \, d\Omega \tag{3.138}$$

Finally, we describe the class of admissible test functions w, which appear in (3.138). Clearly, the values of w and its first derivatives must exist so that the integrals in (3.138) are well-defined. Furthermore, the value of w should vanish on Γ_ϕ, which we previously defined in *Statement of the Stationary Problem* to be that portion of $\partial\Omega$ on which ϕ is specified.

Now, we can completely replace the set of differential equations and interface conditions given in *Governing Equations* with the following alternative problem: Find the function $\phi(\mathbf{x})$, such that $\phi = \phi_b$ on Γ_ϕ, $w = 0$ on Γ_ϕ, and

$$\int_{\Omega} (w\boldsymbol{v} \cdot \boldsymbol{\nabla}\phi - \boldsymbol{\nabla} w \cdot \boldsymbol{f}) \; d\Omega = -\int_{\partial\Omega \setminus \Gamma_\phi} w f_n \, d\Gamma + \int_{\Omega} w\phi_{\mathrm{V}} \, d\Omega \tag{3.139}$$

for all admissible w.

Equation (3.139) is often called a weak statement of the problem given by (3.128) because the second derivatives of the scalar variable ϕ (or alternately, first derivatives of flux $\boldsymbol{f}$) do not appear. More specifically, the original differential equation (3.128) requires that the solution $u(\mathbf{x})$ have second derivatives that exist in each subdomain Ω_ι, whereas (3.139) only requires that the first derivatives of $\phi(\mathbf{x})$ be integrable over the entire domain Ω. We remark that a solution to the original problem (3.128) is also a solution to the weak statement (3.139). However, the converse is not necessarily true: a solution to the weak statement is only a solution to the original problem if it is sufficiently smooth.

Well-posed Problems

In order that a unique solution exists and behaves itself, for both the stationary and transient problems, there are many requirements on the input data that must be satisfied. By input data we mean the supplied domain geometry, initial conditions, boundary conditions, and material coefficients. A fully detailed enumeration of the requirements on the input data is beyond the scope of this manual, and some requirements fall in the realm of research. We have, however, hinted at some of the known requirements at various points in the text, and below in this Section we mention a couple other issues. This is an important topic, and we hope in a future edition to be able to cover it more fully.

As an aside, consider the meaning of the term well-posed problem. We say that a problem is well-posed if there exists a solution, it is unique, and it depends continuously on the data—otherwise the problem is said to be ill-posed. A well-posed problem is not always more physically realistic than an ill-posed one, and many times a well-posed problem may be unrealistic. As for the term stability, it is most often used to mean that the problem is continuous with respect to the data. That is, if we change the problem slightly, the solution changes only slightly.

Regarding specific requirements on the heat source ϕ_{V}, for the stationary problem, in order that the solution exist, a compatibility condition between ϕ_{V} and the applied boundary data must be satisfied. The compatibility condition is a statement of the global conservation principle for Ω. Its exact form depends upon the individual terms which appear in the differential conservation equations in , as well as the applied boundary data [22]. For the transient problem, any function ϕ_{V} that is sufficiently smooth over each subdomain Ω_i is admissible, as long as it is finite and integrable in space and time.

When using a set of boundary conditions that do not specify temperature at any point, the solution is only defined up to an arbitrary constant, i.e., the solution is not unique. Aria may or may not be able to automatically check for this requirement.

Regarding the smoothness of input functions, we must assume that the necessary derivatives of input quantities exist in order that the solution, $\phi(\mathbf{x}, t)$, exists. In other words, there are restrictions that must be imposed on the smoothness of the various input data, including material properties and boundary conditions. The smoothness restrictions are stronger in the strong form of the equations, and correspondingly weaker in the weak form of the equations discussed in this chapter. It is the weak form of the equations to which the Aria program actually tries to provide an

approximate solution. No matter what the smoothness restrictions, Aria cannot reliably enforce them—especially inside user subroutines.

To summarize, it is primarily up to the user to provide valid admissible input data. Exercise caution to ensure a proper problem formulation.

Galerkin Approximation for the Stationary Problem

Solutions to the weak statement given by (3.139) lie in a certain infinite-dimensional space of functions that have derivatives; we denote this space H. Galerkin's method seeks approximate solutions to the weak statement represented by a linear combination of a finite set of basis functions $\{\psi_1, \psi_2, \ldots, \psi_N\}$ that defines a finite-dimensional subspace $H^h \subset H$. We then seek a function $\phi_h \in H^h$ of the form

$$\phi_h = \sum_{i=1}^{N} \alpha_i \psi_i, \tag{3.140}$$

which satisfies the weak form in H^h and where the α_i are unknown constants. If the ψ_i are Lagrange basis functions, then the constants $\alpha_i = \phi_i$: the constants correspond to the evaluation of the approximation at a node.

Upon substitution of (3.140) into (3.139) and allowing the test function w to be each element of H^h, we arrive at the discrete form of our weak statement: Find the function $\phi_h(\mathbf{x}) \in H^h$, such that $\phi_h = \phi_b$ on Γ_ϕ and

$$\begin{aligned} \sum_{j=1}^{N} \left[\int_\Omega \left(\psi_i \boldsymbol{v} \cdot \boldsymbol{\nabla} \psi_j + \boldsymbol{\nabla} \psi_i \cdot k \boldsymbol{\nabla} \psi_j \right) \, d\Omega \right] \alpha_j \\ = - \int_{\partial\Omega \setminus \Gamma_\phi} \psi_i f_n \, d\Gamma + \int_\Omega \psi_i \phi_{\mathrm{V}} d\Omega, \end{aligned} \tag{3.141}$$

for all $\psi_i, i = 1, \ldots, N$. We have assumed a diffusive-style flux such that $\boldsymbol{f} = -k\boldsymbol{\nabla}\phi$,

Here, (3.141) represents N fully discrete equations for the unknown constants ϕ_j. For known flux functions $f_n \neq f_n(u)$, we may write this system of equations as the matrix system

$$\sum_{j=1}^{N} \left(U_{ij} + K_{ij} \right) \alpha_j = f_i, \quad i = 1, \ldots, N \tag{3.142}$$

where

$$U_{ij} = \int_\Omega \psi_i \boldsymbol{v} \cdot \boldsymbol{\nabla} \psi_j \, d\Omega$$

$$K_{ij} = \int_\Omega \boldsymbol{\nabla} \psi_i \cdot k \boldsymbol{\nabla} \psi_j \, d\Omega \tag{3.143}$$

$$f_i = -\int_{\partial\Omega \setminus \Gamma_\phi} \psi_i f_n \, d\Gamma + \int_\Omega \psi_i \phi_V \, d\Omega$$

The matrix U_{ij} represents the convection matrix, K_{ij} the diffusion matrix, and F_i is the forcing vector. Note that if $\mathbf{v} \neq 0$, then the matrix system in (3.142) is non-symmetric.

The finite element method provides a convenient, systematic way to construct the basis functions ψ_i. We address this important issue in *Finite Element Approximation*, in which we also discuss the case where f_n is a function of the temperature, wherein the essential difference is that the associated boundary integral has contributions to the matrix entries K_{ij} as well as the forcing function f_i.

Weak Statement of the Transient Problem

The development of the weak statement for the time-dependent case closely follows that for the stationary case, which we developed in *Weak Statement of the Stationary Problem*. We begin with the time-dependent scalar conservation equation, which we rewrite here as

$$\left(\frac{\partial \phi}{\partial t} + \boldsymbol{v} \cdot \boldsymbol{\nabla} \phi \right) + \boldsymbol{\nabla} \cdot \boldsymbol{f} - \phi_V = 0 \tag{3.144}$$

The key difference between (3.144) and (3.128) is the introduction of the new independent variable, t, and the associated time derivative term. All of the discussion in *Weak Statement of the Stationary Problem* regarding subdomain interfaces, flux jumps, and solution smoothness is relevant for the present case. If the method of weighted residuals is applied to (3.144), for the *Statement of the Transient Problem*, the following weak statement is obtained: Find the function $\phi(\mathbf{x}, t)$, such that $\phi(\mathbf{x}, t_0) = \phi_0$, $\phi(\mathbf{x}, t) = \phi_b$ on Γ_ϕ, and $w = 0$ on Γ_ϕ, which satisfies

$$\int_\Omega \left(w \frac{\partial \phi}{\partial t} + w \boldsymbol{v} \cdot \boldsymbol{\nabla} \phi - \boldsymbol{\nabla} w \cdot \boldsymbol{f} \right) d\Omega = -\int_{\partial\Omega \setminus \Gamma_\phi} w f_n \, d\Gamma + \int_\Omega w \phi_V \, d\Omega \tag{3.145}$$

for all admissible w.

Galerkin Approximation for the Transient Problem

As in *Galerkin Approximation for the Stationary Problem*, let the infinite-dimensional space of solution functions be denoted H. We again introduce a finite set of basis functions $\{\psi_1, \psi_2, \ldots, \psi_N\}$, which defines a finite-dimensional subspace $H^h \subset H$. Given a value of t, we then seek a function $\phi_h(\mathbf{x}, t) \in H^h$ of the form

$$\phi_h(\mathbf{x}, t) = \sum_{i=1}^{N} \alpha_i(t)\psi_i(\mathbf{x}) \tag{3.146}$$

which satisfies the weak form in H^h, and where the $\alpha_i(t)$ are unknown, time-dependent coefficients. If the ψ_i are Lagrange basis functions, then the parameters $\alpha_i(t) = \phi_i(t)$: the interpolation parameters correspond to the evaluation of the interpolation function at a given node. We assume that t is bounded,

$$t_0 \leq t \leq t_1 < \infty$$

Upon substitution of (3.146) into (3.145), and setting $w = \psi_i$, we arrive at the semi-discrete form of our weak statement: Find the function $\phi_h(\mathbf{x}, t) \in H^h$, where $t_0 \leq t \leq t_1$, such that $\phi_0) = \phi_0$, $\phi_h(\mathbf{x}, t) = \phi_b$ on Γ_ϕ, and

$$\sum_{j=1}^{N} \left[\int_\Omega \psi_i \psi_j d\Omega \right] \dot{\alpha}_j(t) \quad + \sum_{j=1}^{N} \left[\int_\Omega \left(\psi_i \boldsymbol{v} \cdot \nabla \psi_j + \nabla \psi_i \cdot k \nabla \psi_j \right) d\Omega \right] \alpha_j(t) = - \int_{\partial\Omega \setminus \Gamma_\phi} \psi_i f_n \, d\Gamma + \int_\Omega \psi_i \phi_\mathrm{V} \, d\Omega \tag{3.147}$$

for each $\psi_i, i = 1, \ldots, N$, and where $\dot{\alpha}(t)$ indicates the time derivative of $\alpha(t)$.

Here, (3.147) represents N ordinary differential equations for the N unknown functions $\alpha_j(t)$. For known flux functions $f_n \neq f_n(u)$, we may write this system of equations as the matrix system

$$\sum_{j=1}^{N} M_{ij} \dot{\alpha}_j(t) + \sum_{j=1}^{N} \left(U_{ij} + K_{ij} \right) \alpha_j(t) = f_i, \quad i = 1, \ldots, N \tag{3.148}$$

where

$$\begin{aligned} M_{ij} &= \int_\Omega \psi_i \psi_j d\Omega \\ U_{ij} &= \int_\Omega \psi_i \boldsymbol{v} \cdot \boldsymbol{\nabla} \psi_j \, d\Omega \\ K_{ij} &= \int_\Omega \boldsymbol{\nabla} \psi_i \cdot k \boldsymbol{\nabla} \psi_j \, d\Omega \\ f_i &= - \int_{\partial\Omega \setminus \Gamma_\phi} \psi_i f_n \, d\Gamma + \int_\Omega \psi_i \phi_\mathrm{V} \, d\Omega \end{aligned} \tag{3.149}$$

The matrix M_{ij} is often referred to as the mass or capacitance matrix. Note that if $\mathbf{v} \neq 0$, then the matrix system in (3.148) is non-symmetric. Also, if any of the parameters in (3.149), such as the material properties, applied fluxes, or volumetric heating are time dependent, then the corresponding matrices or vectors will also be time dependent. The system of ordinary differential equations given in (3.148) must be numerically integrated in time. We discuss this issue in *Time Discretization*.

Mass Lumping

In contrast with the consistent mass formulation, a lumping procedure instead uses the nodally evaluated quantity effectively "lumping" the contribution, i.e., the first term in (3.148) becomes

$$\dot{\alpha}_i(t) \sum_{j=1}^{N} M_{ij} \tag{3.150}$$

Mass lumping can also be applied to source terms, or boundary fluxes via an analogies procedure.

Time Discretization

There are many ways to integrate the system of ordinary differential equations given in (3.148). The method employed by Aria uses is usually referred to as the variable-implicit method or theta-method. Aria has the ability to adaptively step the ODE to reduce local error with respect to time, but we emphasize that this says nothing about the global error in the ODE, i.e., the tolerance on the local error is not a tolerance on the total accuracy of the ODE solution. More significantly, we should point out that a small error in the temporal discretization does not necessarily imply that there is a small error in the spatial discretization.

We begin the description of Aria's time integration strategy by rewriting (3.148) as

$$\mathbf{M}(t)\dot{\boldsymbol{\phi}}(t) + \hat{\mathbf{K}}(t)\boldsymbol{\phi}(t) - \mathbf{f}(t) = 0, \tag{3.151}$$

where $\dot{\boldsymbol{\phi}}(t)$ and $\boldsymbol{\phi}(t)$ are the vectors of entries $\dot{\alpha}_j(t)$ and $\alpha_j(t)$, respectively, and where $\mathbf{M}$ is the matrix with entries given by (3.149), $\hat{\mathbf{K}}$ is the matrix with entries $\hat{K}_{ij} = U_{ij} + K_{ij}$, and $\mathbf{f}$ is the vector with entries given by (3.149).

Numerical time integration of a partial differential equation requires both a discretized form of the governing equation and a time integrator. The discrete form of (3.151) for time level n can be written as

$$\mathbf{M}^n\dot{\boldsymbol{\phi}}^n + \hat{\mathbf{K}}^n\boldsymbol{\phi}^n - \mathbf{f}^n = 0. \tag{3.152}$$

A time integrator requires that the discrete approximation of $\boldsymbol{\phi}(t)$ be expressed as a function of calculated data and one choice of this approximation is

$$\boldsymbol{\phi}^{n+1} = f(\boldsymbol{\phi}^{n+1}, t^{n+1}, \boldsymbol{\phi}^n, t^n, \boldsymbol{\phi}^{n-1}, t^{n-1}, \ldots) \tag{3.153}$$

where f is often a function of computed derivatives of $\boldsymbol{\phi}$.

One form of (3.153) used in Aria is an implicit method which employs

$$\boldsymbol{\phi}^{n+1} = \boldsymbol{\phi}^n + \Delta t(1-\theta)\dot{\boldsymbol{\phi}}^n + \theta\Delta t\dot{\boldsymbol{\phi}}^{n+1} \tag{3.154}$$

where Δt is the time step size and θ denotes the evaluation level within the time step, $\theta = 0 \Rightarrow t(n)$ and $\theta = 1 \Rightarrow t(n+1)$.

The degree of implicitness of a given time discretization of (3.151) is determined by the time level at which $\hat{\mathbf{K}}(t)\boldsymbol{\phi}(t)$ is evaluated—if at the old time level, then the scheme is said to be explicit, if at the new time level, then fully implicit, and if at some average of the two, then semi-implicit. Combining (3.154) with (3.152) at time levels n and $n+1$ one obtains a variable implicit method

$$\left(\frac{1}{\Delta t^n}\mathbf{M} + \theta\hat{\mathbf{K}}^{n+1}\right)\boldsymbol{\phi}^{n+1} = \theta\mathbf{f}^{n+1} + (1-\theta)\mathbf{M}\dot{\boldsymbol{\phi}}^n + \frac{1}{\Delta t^n}\mathbf{M}\boldsymbol{\phi}^n \tag{3.155}$$

The case of $\theta = 1$ corresponds to Euler implicit, $\theta = 1/2$ corresponds to the trapezoid rule, and $\theta = 2/3$ corresponds to Galerkin implicit. In general, this method is accurate only to first order in time, but the special case of $\theta = 1/2$ can be shown to be to be second-order accurate. It is important to note that if $\hat{\mathbf{K}}$ or $\mathbf{f}$ depend upon $\boldsymbol{\phi}$, then (3.155).

The time derivative $\dot{\boldsymbol{\phi}}^n$ can be calculated using (3.154) as

$$\dot{\boldsymbol{\phi}}^n = \frac{\boldsymbol{\phi}^n - \boldsymbol{\phi}^{n-1}}{\theta\Delta t^{n-1}} - \frac{1-\theta}{\theta}\dot{\boldsymbol{\phi}}^{n-1} \ . \tag{3.156}$$

Note that for $\theta = 1$, $\dot{\boldsymbol{\phi}}^n$ is not needed for evaluation of (3.155). If $\theta \neq 1$, then the recursive nature of (3.156) introduces some difficulty to the solution scheme since $\dot{\boldsymbol{\phi}}^{n-1}$ is not initially known. To circumvent this potential issue, Aria uses the fully-implicit case ($\theta = 1$) and (3.155) becomes

$$\left(\frac{1}{\Delta t^n}\mathbf{M} + \hat{\mathbf{K}}^{n+1}\right)\boldsymbol{\phi}^{n+1} = \mathbf{f}^{n+1} + \frac{1}{\Delta t^n}\mathbf{M}\boldsymbol{\phi}^n \tag{3.157}$$

or as one might expect

$$\frac{1}{\Delta t^n}\mathbf{M}\dot{\boldsymbol{\phi}}^{n+1} + \hat{\mathbf{K}}^{n+1}\boldsymbol{\phi}^{n+1} - \mathbf{f}^{n+1} \ .$$

Thus in the fully-implicit formulation one solves the discretized set of equation using information from the current time plane with a selected representation of the time derivative.

In the simplest case one begins with an initial guess for $\boldsymbol{\phi}^n$ and uses the first-order accurate time derivative known as BDF1 (Backward Finite Difference 1).

$$\dot{\boldsymbol{\phi}}^{n+1} = \frac{\boldsymbol{\phi}^{n+1} - \boldsymbol{\phi}^n}{\Delta t^n} \tag{3.158}$$

Another choice for the time derivative $\dot{\boldsymbol{\phi}}^n$ is the second-order approximation known as BDF2 (Backward Finite Difference 2)

$$\dot{\boldsymbol{\phi}}^{n+1} = \frac{3\boldsymbol{\phi}^{n+1} - 4\boldsymbol{\phi}^n + \boldsymbol{\phi}^{n-1}}{2\Delta t} \ . \tag{3.159}$$

Obviously utilization of the above requires that the first two solution steps of (3.157) be obtained using the first-order derivative (3.158). Theoretical time step bounds are available for the BDF2 time integrator and the scheme is often used in time accurate simulations.

While it is possible to use a fixed, constant Δt while iterating (3.157), it is often preferable to allow the timestep size to vary. Aria uses an automatic timestep size selection algorithm that allows Δt to vary while satisfying several constraints[3]. The approach first calculates a candidate timestep size, using local time truncation error estimates, then adjusts this value using several heuristic rules.

To calculate the candidate timestep size, assume that two time integrators of comparable accuracy are available. If we compare local truncation error estimates for each scheme, and if we have a target value of a truncation error norm in mind, then we can estimate Δt. The implicit scheme given by (3.155) is one such integrator, but a second is needed. Since this second scheme is only required for the time step selection algorithm, computational speed is important, and therefore we consider only explicit schemes. If θ is chosen so that the implicit scheme is first-order accurate in time, then we use forward Euler differencing, which may be written as

$$\boldsymbol{\phi}_p^{n+1} = \boldsymbol{\phi}^n + \Delta t^n \dot{\boldsymbol{\phi}}^n \tag{3.160}$$

where $\boldsymbol{\phi}_p$ indicates the temperature that is predicted as a result of applying (3.160), and the acceleration vector $\dot{\boldsymbol{\phi}}^n$ is calculated using (3.156). For $\theta = 1/2$, a second-order accurate explicit scheme is required, and the Adams–Bashforth difference formula is used, namely

$$\boldsymbol{\phi}_p^{n+1} = \boldsymbol{\phi}^n + \frac{\Delta t^n}{2}\left[\left(2 + \frac{\Delta t^n}{\Delta t^{n-1}}\right)\dot{\boldsymbol{\phi}}^n - \frac{\Delta t^n}{\Delta t^{n-1}}\dot{\boldsymbol{\phi}}^{n-1}\right] \tag{3.161}$$

where the acceleration vectors are calculated using $\theta = 1/2$ in (3.156).

For $\theta > 1/2$, the candidate time step size, $(\Delta t)_c$, is estimated according to

$$(\Delta t)_c^{n+1} = \Delta t^n \sqrt{\frac{\theta\varepsilon}{(\theta - 1/2)\|\boldsymbol{\phi}^{n+1} - \boldsymbol{\phi}_p^{n+1}\|}} \tag{3.162}$$

where ε is the given, allowable value of the truncation error norm[4]. A typical value of the truncation error norm would be 0.0001. Here, the norm $\|\cdot\|$ indicates the non-dimensional root mean square norm of a vector,

$$\|\mathbf{v}\| = \frac{1}{v_{\max}}\sqrt{\frac{1}{N}\sum_{i=1}^{N} v_i^2} \tag{3.163}$$

where

$$v_{\max} = \max_{1 \le i \le N} |v_i|$$

[3] This algorithm is based on the approached described by Gresho, et al. [25] and is similar to that which is implemented in COYOTE [26] [27].

[4] We do not derive the local truncation error estimates here, since the method of using Taylor series analysis for this purpose is well established, for example [25].

Note that in (3.162), the timestep grows as the square root of ε. For $\theta = 1/2$, the candidate timestep size is calculated according to the second-order formula

$$(\Delta t)_c^{n+1} = \Delta t^n \sqrt[3]{3\left(1 + \frac{\Delta t^{n-1}}{\Delta t^n}\right) \frac{\varepsilon}{\|\boldsymbol{\phi}^{n+1} - \boldsymbol{\phi}_p^{n+1}\|}} \tag{3.164}$$

Note that in (3.164), the timestep grows with the cube root of ε. We emphasize that the initialization problem caused by the recursive nature of (3.161) is circumvented by using the initial Δt with $\theta = 1$ for the first two timesteps, then switching to $\theta = 1/2$ and using (3.164) to calculate Δt thereafter.

In practice, it is useful to limit the candidate timestep size calculated by the above procedure. Aria imposes three limits on the timestep. The first limit is enforced only when there is volumetric heating due to chemical reactions, as their effect is not directly included in either (3.162) or (3.164). Aria uses the CHEMEQ library [23] for time integration of chemistry. CHEMEQ uses a stiff ordinary differential equation integrator, which adaptively subcycles species equations according to their stiffness. Usually, the chemical time scales are much shorter than that of conduction. Hence, if $(\delta t)_s^n$ represents the minimum timestep size used to integrate the chemical species at timestep n, then we limit $(\Delta t)_c^{n+1}$ according to

$$(\Delta t)_c^{n+1} = \min\left((\Delta t)_c^{n+1}, \chi(\delta t)_s^n\right) \tag{3.165}$$

where χ is a multiplication factor, typically on the order of 100.

The second limit on the timestep allows control of how fast the solution changes from one timestep to the next. Let $|\Delta u|_m$ denote the maximum change in temperature at a single mesh node from timestep n to $n+1$, and $|\Delta T|_a$ denote the allowable value of this difference. Then we enforce

$$\text{if} \quad \left(|\Delta T|_m \frac{(\Delta t)_c^{n+1}}{\Delta t^n} > |\Delta T|_a\right) \quad \text{then} \quad (\Delta t)_c^{n+1} = \frac{0.95|\Delta T|_a \Delta t^n}{|\Delta T|_m} \tag{3.166}$$

Equation (3.166) states that if the estimated maximum change in temperature exceeds the allowable change, then the candidate timestep size is set to 5% less than the value required to prevent this condition.

Finally, the user is allowed to specify minimum and maximum values of the timestep size, $\Delta t_{\min}$ and $\Delta t_{\max}$, respectively. For the BDF2 method a theoretical limit of $\Delta t_{\max} = (1 + \sqrt{2})\Delta t^n$ is also imposed. Therefore, we also restrict the candidate timestep size so that

$$\Delta t_{\min} \leq (\Delta t)_c^{n+1} \leq \Delta t_{\max} \tag{3.167}$$

In summary, the candidate timestep size is first estimated using (3.162) for $\theta > 1/2$ and (3.164) for $\theta = 1/2$. Next, each of the applicable limits given in (3.165)–(3.167) are enforced to determine the timestep size for the next iteration of (3.155), Δt^{n+1}.

3.9 Finite Element Approximation

In *Discretization*, we discretized the governing equations in space using Galerkin's method. In this chapter, we discuss two important, unresolved issues. The first issue is the method for constructing the basis functions ψ, which appear in the weak statements given by *the stationary weak description* and *the transient weak description*. The second issue is the calculation of the associated integrals over the domain and its boundary. The finite element method solves both of these issues, and is concisely described by Becker, et al [22]:

> The finite element method provides a general and systematic technique for constructing basis functions for Galerkin approximations of boundary value problems. The main idea is that the basis functions [. . .] can be defined piecewise over subregions of the domain called *finite elements* and that over any subdomain the [basis functions] can be chosen to be very simple functions such as polynomials of low degree.

If the finite elements are also chosen to have convenient shapes (like triangles and quadrilaterals) for calculating integrals, then the integration over the entire domain is also facilitated.

To elaborate, consider Fig. 3.10, which shows a one-dimensional domain partitioned into four finite elements. In Fig. 3.11, the global, piecewise linear basis function for node i, ψ_i, is shown as the standard hat function. This function has a value of unity at node i and zero in elements not connected to node i. The finite element approach to constructing ψ_i is shown in Fig. 3.11. On each element, two linear functions ψ_i^e are defined, $e \in \{1, 2, 3, 4\}$, $i \in \{1, 2\}$. If we add ψ_2^2 from the left element to ψ_1^3 from the right element, then we see that the global basis function ψ is constructed.

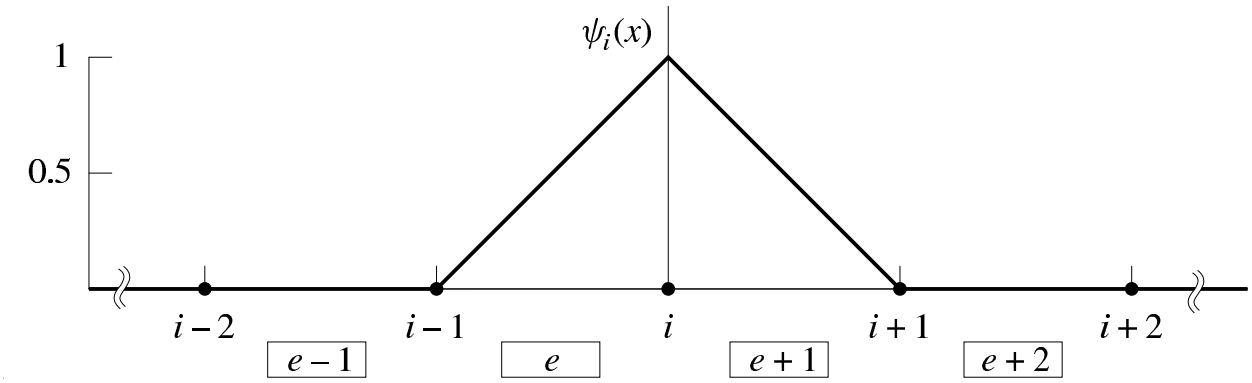


Fig. 3.10: A linear finite element global basis function at a node is formed from the nodal shape functions of two neighboring elements. The global basis function for node i has support over the two elements e and $e+1$ that share the node, and takes on the value of zero everywhere else.

Locally, the element's shape functions can be used to interpolate values defined at the nodes to arbitrary location within the element. Specifically, for some quantity q defined at the N (local) nodes of the element e, the value of q within the element is

$$q(\boldsymbol{x}, t) = \sum_{i=1}^{N} q_i(t)\psi_i^e(\boldsymbol{x})$$

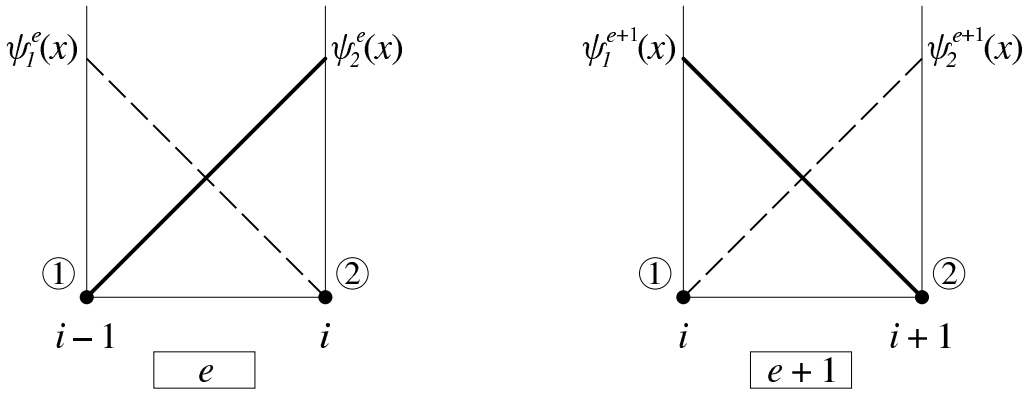


Fig. 3.11: The associated element shape functions, ψ_2^e (on local node 2 of element e), and ψ_1^{e+1} (on local node 1 of element $e+1$), are combined to define the global basis function for node i.

Now, consider the global stiffness matrix $\mathbf{K}$, with row i and column j, *defined in the previous section*. We now rewrite this integral as the sum over each finite element, namely

$$K_{ij} = \sum_{e=1}^{E} \int_{\Omega^e} \nabla\psi_i \cdot \mathbf{k}\nabla\psi_j \, d\Omega \tag{3.168}$$

In (3.168), the global basis function ψ_i is nonzero only on those elements which support node i. Hence we can consider the contributions to the global stiffness matrix from element e, and define

$$K_{ij}^e = \int_{\Omega^e} \nabla\psi_i^e \cdot \mathbf{k}\nabla\psi_j^e \, d\Omega \tag{3.169}$$

There is a subtle difference in the interpretation of the indices in (3.168) as opposed to (3.169). In (3.168), the indices i and j are global, and span all nodes in the finite element mesh. In (3.169), the indices i and j are local, and only span the number of element shape functions. Aria calculates the element stiffness matrix contributions defined in (3.169) and assembles the results into the correct locations in the global stiffness matrix using a mapping from the local node numbers of a given element to the associated global node numbers. In fact, each of the integrals in the weak statements derived in *Discretization* involves a basis function and is calculated in the same way.

In the following section, *Element Integration*, we discuss how the integrals are computed on a master element. Next, in *Linear Master Elements*, we discuss the master element interpolation functions for the elements used in Aria.

Element Integration

Element integrals of the form given by (3.169) are defined on the element coordinates in physical space. When calculating these quantities numerically, it is convenient to transform these integrals from physical space to a reference element, also called a master element, in parametric space. This transformation is performed locally, for each finite element in the mesh. In this section we discuss the details of this process for two-dimensional physical space in Cartesian coordinates. The approach is completely general, and extensible to three-dimensional space and other coordinate systems. This situation is illustrated in Fig. 3.12, showing such a transformation from the quadrilateral element, which spans the region $[-1, 1] \times [-1, 1]$, in parametric coordinates (ξ, η) to an arbitrary region in physical coordinates (x, y).

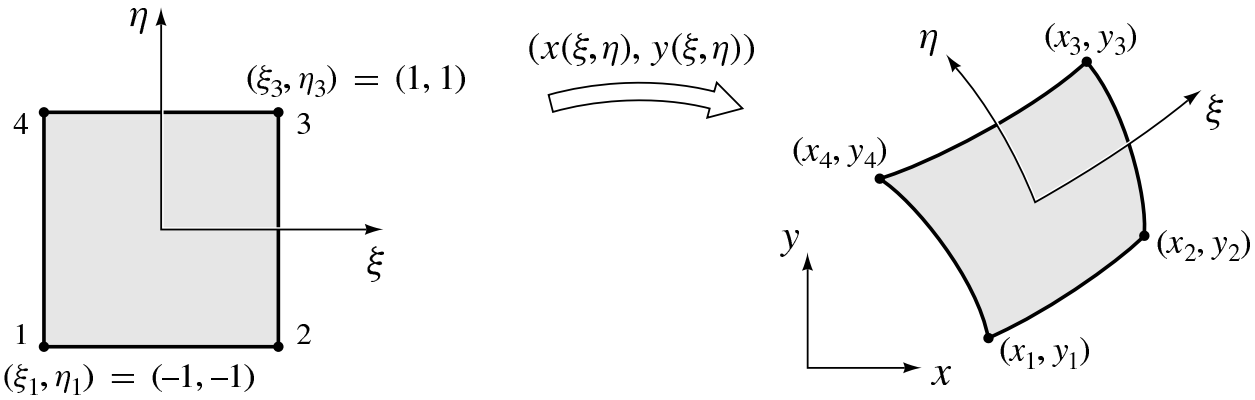


Fig. 3.12: Mapping from the master element to physical coordinates: The quadrilateral master element in (ξ, η) on the region $[-1, 1] \times [-1, 1]$ is mapped to its image in physical coordinates (x, y).

Returning to (3.169), given some function $f(x, y)$, we wish to calculate the integral

$$I = \int_{\Omega^e} f(x, y)\, dx\, dy \tag{3.170}$$

on the master element in (ξ, η) space. The change of variables theorem [28] from vector calculus states that (3.170) is equivalent to

$$I = \int_{\hat{\Omega}^e} \hat{f}(\xi, \eta)|\mathbf{J}|\, d\xi\, d\eta \tag{3.171}$$

where $\hat{\Omega}^e$ is the master element and $|\mathbf{J}|$ is the determinant of the non-singular Jacobian matrix of the transformation

$$\begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} x(\xi, \eta) \\ y(\xi, \eta) \end{pmatrix} \tag{3.172}$$

Recall the Jacobian matrix of (3.172)

$$\mathbf{J} = \begin{pmatrix} \dfrac{\partial x}{\partial \xi} & \dfrac{\partial x}{\partial \eta} \\ \dfrac{\partial y}{\partial \xi} & \dfrac{\partial y}{\partial \eta} \end{pmatrix} \tag{3.173}$$

and its determinant

$$|\mathbf{J}| = \frac{\partial x}{\partial \xi}\frac{\partial y}{\partial \eta} - \frac{\partial x}{\partial \eta}\frac{\partial y}{\partial \xi} \tag{3.174}$$

Using the local finite element shape functions to construct isoparametric transformations for (3.172), we have

$$\begin{aligned} x(\xi,\eta) &= \sum_{i=1}^{n} x_i \hat{\psi}_i^e \\ y(\xi,\eta) &= \sum_{i=1}^{n} y_i \hat{\psi}_i^e, \end{aligned} \tag{3.175}$$

where n is the number of shape functions on element e, and the coordinates of each node of the element is given by the pair (x_i, y_i).

Since we seek a numerical solution, it is convenient to compute the integrals numerically; we use Gaussian quadrature rules of the form

$$I = \int_{\hat{\Omega}^e} \hat{f}(\xi,\eta)|\mathbf{J}|\, d\xi\, d\eta \approx \sum_{g=1}^{G} w_g \hat{f}(\xi_g,\eta_g)|\mathbf{J}|_{(\xi_g,\eta_g)}, \tag{3.176}$$

where the number of Gauss points is denoted G, (ξ_g, η_g) are the Gauss point coordinates (abscissas), $|\mathbf{J}|_{(\xi_g,\eta_g)}$ indicates that the Jacobian determinant is to be evaluated at the Gauss point coordinates, and w_g is the associated weight. For master element shapes in common use, such as triangles, hexahedra, quadrilaterals, etc., the quadrature points and weights are well-known and tabulated, e.g., see [22].

Table 3.1: Weights and coordinates for the Gauss quadrature rules on elements with linear shape functions.

element topology	degree G	weights w_g	coordinates ξ_g
1D Edge	2	$1, 1$	$-\frac{1}{\sqrt{3}}, \frac{1}{\sqrt{3}}$
2D Triangle	3	$\frac{1}{6}, \frac{1}{6}, \frac{1}{6}$	$(\frac{1}{2}, \frac{1}{2}), (0, \frac{1}{2}), (\frac{1}{2}, 0)$
2D Quadrilateral	4	$\frac{1}{4}, \frac{1}{4}, \frac{1}{4}, \frac{1}{4}$	$(-\gamma, -\gamma), (\gamma, -\gamma),$ $(\gamma, \gamma), (-\gamma, \gamma)$
3D Tetrahedron	4	$\frac{1}{24}, \frac{1}{24}, \frac{1}{24}, \frac{1}{24}$	$(\alpha, \beta, \beta), (\beta, \alpha, \beta),$ $(\beta, \beta, \alpha), (\beta, \beta, \beta)$
3D Hexahedron	8	$\frac{1}{8}, \frac{1}{8}, \frac{1}{8}, \frac{1}{8}, \frac{1}{8}, \frac{1}{8}, \frac{1}{8}, \frac{1}{8}$	$(-\gamma, -\gamma, -\gamma), (\gamma, -\gamma, -\gamma),$ $(\gamma, \gamma, -\gamma), (-\gamma, \gamma, -\gamma),$ $(-\gamma, -\gamma, \gamma), (\gamma, -\gamma, \gamma),$ $(\gamma, \gamma, \gamma), (-\gamma, \gamma, \gamma)$

where $\gamma = \sqrt{3}/6$, $\alpha = 0.58541020$ and $\beta = 0.13819660$. For linear elements, Aria uses the rules shown in Table 3.1, which correspond to the master element topologies given in *Linear Master Elements*. Because the order of quadrature is fixed for a given element (and the associated polynomial degree of element shape functions), the user of Aria should not forget to account for possible errors in the quadrature. Error in quadrature could be significant when user subroutines for boundary conditions, source terms, etc., attempt to represent highly oscillatory, or unsmooth data.

Linear Master Elements

In this section, we present the shape functions $\hat{\psi}_i^e$ for the linear master elements currently supported in Aria. Presently, all elements use the Lagrange basis functions. For the sake of brevity, we drop the superscript e on the element shape function. Hence, the notation $\hat{\psi}_i$ is equivalent to $\hat{\psi}_i^e$.

2D Triangular Element

The master element for the two-dimensional triangle with nodes numbered locally as shown. The Gauss quadrature rule given in Table 3.1 for the 2D Triangle is used to integrate this element. The shape functions are

$$\begin{aligned} \hat{\psi}_1 &= 1 - \xi - \eta \\ \hat{\psi}_2 &= \xi \\ \hat{\psi}_3 &= \eta \end{aligned} \tag{3.177}$$

2D Quadrilateral Element

The master element for the two-dimensional quadrilateral is shown in Fig. 3.12. The Gauss quadrature rule for the 2D Quadrilateral given in Table 3.1 is used to integrate this element. The shape functions are

$$\begin{aligned} \hat{\psi}_1 &= \frac{1}{4}(1-\xi)(1-\eta) \\ \hat{\psi}_2 &= \frac{1}{4}(1+\xi)(1-\eta) \\ \hat{\psi}_3 &= \frac{1}{4}(1+\xi)(1+\eta) \\ \hat{\psi}_4 &= \frac{1}{4}(1-\xi)(1+\eta) \end{aligned} \tag{3.178}$$

2D Face Element

When applying flux boundary conditions, it is often necessary to integrate on the surfaces of the two-dimensional elements presented in *2D Triangular Element* and *2D Quadrilateral Element*. In both cases, the surface is topologically a one-dimensional edge. The two shape functions that define this element are

$$\begin{aligned} \hat{\psi}_1 &= \frac{1}{2}(1-\xi) \\ \hat{\psi}_2 &= \frac{1}{2}(1+\xi) \end{aligned} \tag{3.179}$$

The Gauss quadrature rule for the 1D Edge topology that is given in Table 3.1 is used to integrate this element.

Note that the location along this edge is given by the single parametric coordinate, ξ, although there is a pair of physical Cartesian coordinates, (x, y) associated with each value of ξ. This fact makes the construction of the transformation given as (3.172) a little less obvious. In this case, the

determinant of the Jacobian matrix is the ratio of the edge length in physical coordinates to the edge length in computational coordinates,

$$|\mathbf{J}| = \sqrt{\left(\frac{\partial x}{\partial \xi}\right)^2 + \left(\frac{\partial y}{\partial \xi}\right)^2}$$

Since we are using isoparametric mappings, it is easy to show that, using the linear shape functions given as (3.179), this determinant is given by

$$|\mathbf{J}| = \frac{1}{2}\sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$$

3D Tetrahedron Element

The topology of the four-node, linear tetrahedron element is shown in Fig. 3.13.

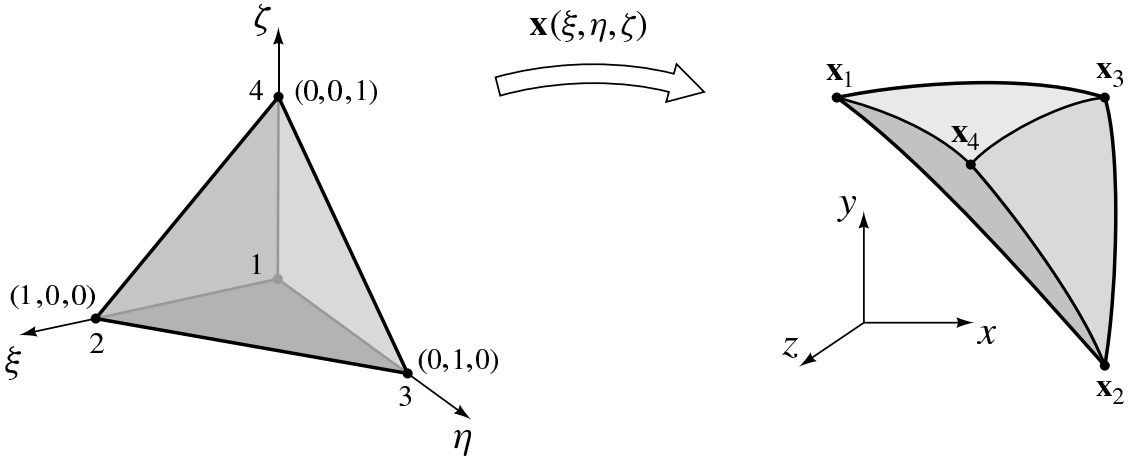


Fig. 3.13: The master tetrahedron element is mapped to its image in physical coordinates (x, y, z)**.**

The Gauss quadrature rule for the 3D Tetrahedron given in Table 3.1 is used to integrate this element. The shape functions are

$$\begin{aligned} \hat{\psi}_1 &= 1 - \xi - \eta - \zeta \\ \hat{\psi}_2 &= \xi \\ \hat{\psi}_3 &= \eta \\ \hat{\psi}_4 &= \zeta \end{aligned} \tag{3.180}$$

3D Triangular Face Element

The tetrahedral element presented in *3D Tetrahedron Element* has three-node, triangular faces, which are topologically equivalent to the two-dimensional triangular element that was presented in *2D Triangular Element*. Accordingly, the shape functions are identical (3.177). The Gauss quadrature rule for the 2D Triangle that is given in Table 3.1 is used to integrate this element.

Since the physical coordinates are three-dimensional, but there are only two parameters on the surface, (ξ, η), the transformation from computational coordinates to physical coordinates requires a little elaboration. Vector calculus provides a formula for the integral of a scalar function over a parameterized surface [28]. We are interested in the surface integral

$$I = \int_{\Gamma^e} f(x, y, z)\, dx\, dy\, dz \tag{3.181}$$

where Γ^e is the three dimensional surface of our face element, and is given by the isoparametric mapping

$$\begin{pmatrix} x(\xi,\eta) \\ y(\xi,\eta) \\ z(\xi,\eta) \end{pmatrix} = \begin{pmatrix} \sum_{i=1}^{3} x_i \hat{\psi}_i(\xi,\eta) \\ \sum_{i=1}^{3} y_i \hat{\psi}_i(\xi,\eta) \\ \sum_{i=1}^{3} z_i \hat{\psi}_i(\xi,\eta) \end{pmatrix} \tag{3.182}$$

It can be shown that (3.181) may be computed as

$$I = \int_{\hat{\Gamma}^e} \hat{f}(\xi,\eta) |\mathbf{J}|\, d\xi\, d\eta, \tag{3.183}$$

where $\hat{\Gamma}^e$ is the master element, $\hat{f}(\xi,\eta) = f(x(\xi,\eta), y(\xi,\eta), z(\xi,\eta))$, and the determinant of the Jacobian matrix may be written as

$$|\mathbf{J}| = \sqrt{\left|\frac{\partial(x,y)}{\partial(\xi,\eta)}\right|^2 + \left|\frac{\partial(y,z)}{\partial(\xi,\eta)}\right|^2 + \left|\frac{\partial(x,z)}{\partial(\xi,\eta)}\right|^2} \tag{3.184}$$

Equation (3.184) involves the calculation of the determinants of three sub-matrices. For example,

$$\left|\frac{\partial(x,y)}{\partial(\xi,\eta)}\right| = \frac{\partial x}{\partial \xi}\frac{\partial y}{\partial \eta} - \frac{\partial x}{\partial \eta}\frac{\partial y}{\partial \xi}$$

3D Hexahedron Element

Fig. 3.14 shows the topology of the eight-node, linear hexahedral master element.

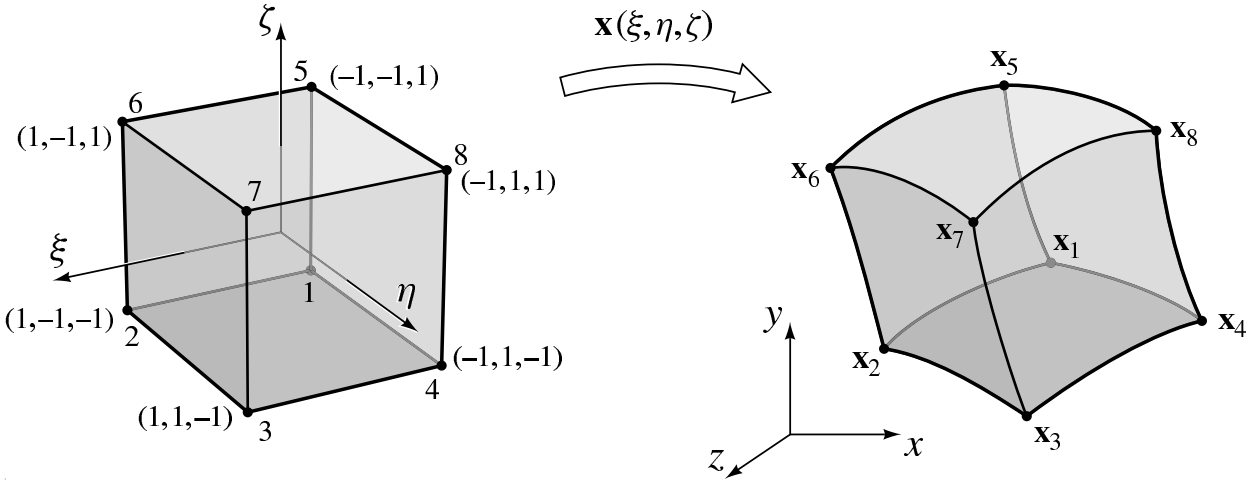


Fig. 3.14: The master hexahedron element in (ξ, η, ζ) on the region $[-1, 1] \times [-1, 1] \times [-1, 1]$ is mapped to its image in physical coordinates (x, y, z).

The Gauss quadrature rule for the 3D Hexahedron that is given in Table 3.1 is used to integrate this element. The shape functions are

$$
\begin{aligned}
\hat{\psi}_1 &= \frac{1}{8}(1-\xi)(1-\eta)(1-\zeta) \\
\hat{\psi}_2 &= \frac{1}{8}(1+\xi)(1-\eta)(1-\zeta) \\
\hat{\psi}_3 &= \frac{1}{8}(1+\xi)(1+\eta)(1-\zeta) \\
\hat{\psi}_4 &= \frac{1}{8}(1-\xi)(1+\eta)(1-\zeta) \\
\hat{\psi}_5 &= \frac{1}{8}(1-\xi)(1-\eta)(1+\zeta) \\
\hat{\psi}_6 &= \frac{1}{8}(1+\xi)(1-\eta)(1+\zeta) \\
\hat{\psi}_7 &= \frac{1}{8}(1+\xi)(1+\eta)(1+\zeta) \\
\hat{\psi}_8 &= \frac{1}{8}(1-\xi)(1+\eta)(1+\zeta)
\end{aligned}
$$

3D Quadrilateral Face Element

The hexahedron element described in *3D Hexahedron Element* has quadrilateral faces. The shape functions that are used for interpolating on one of these faces are identical to the shape functions used for the *2D Quadrilateral Element* in (3.178). As discussed in *3D Triangular Face Element*, integrals on this surface master element may be computed according to (3.183), with the determinant given by (3.184). The Gauss quadrature rule for the 2D Quadrilateral that is given in Table 3.1 is used to integrate this element.

3D Triangular Shell Element

Beta Capability
Shell support in Aria is currently a beta feature. This capability is a work-in-progress and is not thoroughly tested and should be used with caution.

Sometimes, a subdomain $\Omega_i \subset \Omega$ may be very thin, and may also possibly consist of a highly conductive material like aluminum. In this case, it is reasonable to neglect the thermal gradient through the thickness of Ω_i. Shell elements are specialized volume elements that are often used to model this situation. The topology of the linear triangular shell is illustrated in Fig. 3.15, which has three nodes, and two faces. Topologically, the shell has no thickness that can be derived solely from its node coordinates; instead, the thickness is an attribute specified as input data. The shape functions are identical to (3.177) in *2D Triangular Element*.

The transformation from computational coordinates to physical coordinates for the three-dimensional shell element depends on the thickness attribute. In this case, we wish to calculate the volume integral

$$I = \int_{\Omega^e} f(x, y, z)\, dx\, dy\, dz \tag{3.185}$$

The parametric mapping from computational coordinates to physical coordinates may be regarded as the surface mapping, plus a component that comes up out of the shell mid-plane. With reference to Fig. 3.15, we may write this mapping as

$$\begin{pmatrix} x(\xi,\eta,\zeta) \\ y(\xi,\eta,\zeta) \\ z(\xi,\eta,\zeta) \end{pmatrix} = \begin{pmatrix} \sum_{i=1}^{I} x_i \hat{\psi}_i(\xi,\eta) + \zeta\theta_x \sum_{i=1}^{I} \tau_i \hat{\psi}_i(\xi,\eta) \\ \sum_{i=1}^{I} y_i \hat{\psi}_i(\xi,\eta) + \zeta\theta_y \sum_{i=1}^{I} \tau_i \hat{\psi}_i(\xi,\eta) \\ \sum_{i=1}^{I} z_i \hat{\psi}_i(\xi,\eta) + \zeta\theta_z \sum_{i=1}^{I} \tau_i \hat{\psi}_i(\xi,\eta) \end{pmatrix} \tag{3.186}$$

where the shape functions $\hat{\psi}_i$ are given by (3.177), $I = 3$, τ_i denotes the thickness of the shell at node i, $\zeta \in [-1, 1]$ is the parametric coordinate in the direction locally orthogonal to the shell

mid-plane, and we have defined the inner products

$$\begin{aligned} \theta_x &= \hat{\mathbf{n}}_\zeta \cdot \hat{\mathbf{i}} \\ \theta_y &= \hat{\mathbf{n}}_\zeta \cdot \hat{\mathbf{j}} \\ \theta_z &= \hat{\mathbf{n}}_\zeta \cdot \hat{\mathbf{k}} \end{aligned} \tag{3.187}$$

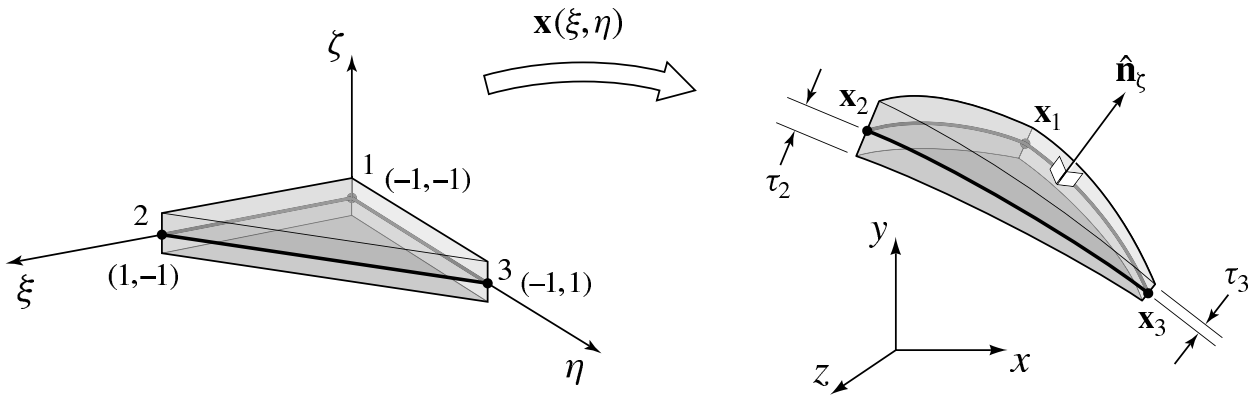


Fig. 3.15: The midplane of the master triangular shell element is mapped to its image in physical coordinates (x, y, z), and the variation in the shell thickness is represented with the element shape functions.

In (3.187), $\hat{\mathbf{n}}_\zeta$ is the unit vector in the direction locally orthogonal to the element mid-plane, and $\hat{\mathbf{i}}, \hat{\mathbf{j}}$ and $\hat{\mathbf{k}}$ are the unit vectors in the x, y and z coordinate directions, respectively. The volume integral in (3.185) on the master element is

$$I = \int_{\hat{\Omega}^e} \hat{f}(\xi, \eta, \zeta) |\mathbf{J}| \, d\xi \, d\eta \, d\zeta \tag{3.188}$$

where $|\mathbf{J}|$ is the determinant of the Jacobian matrix obtained from the transformation given by (3.186).

Aria currently only supports constant thickness shells. In this case, I reduces to

$$I = \tau \int_{\hat{\Gamma}^e} \hat{f}(\xi, \eta, 0) |\mathbf{J}| \, d\xi \, d\eta \tag{3.189}$$

where τ is the constant shell thickness, $\hat{\Gamma}^e$ denotes the triangular surface of the shell mid-plane, $|\mathbf{J}|$ is the determinant of the transformation given by (3.182), and the Gauss quadrature rule for the 2D Triangle that is given in Table 3.1 is used to perform the numerical integration.

3D Quadrilateral Shell Element

Beta Capability
Shell support in Aria is currently a beta feature. This capability is a work-in-progress and is not thoroughly tested and should be used with caution.

The three-dimensional, linear, quadrilateral shell element is, in principle, no different from the triangular shell element which we discussed in *3D Triangular Shell Element*. As illustrated in Fig. 3.16, the quadrilateral shell has four nodes, instead of three, with the basis functions given in (3.178). The transformation from computational coordinates to physical coordinates is again given by (3.186), but with $I = 4$, and the $\hat{\psi}_i$ are defined by (3.178).

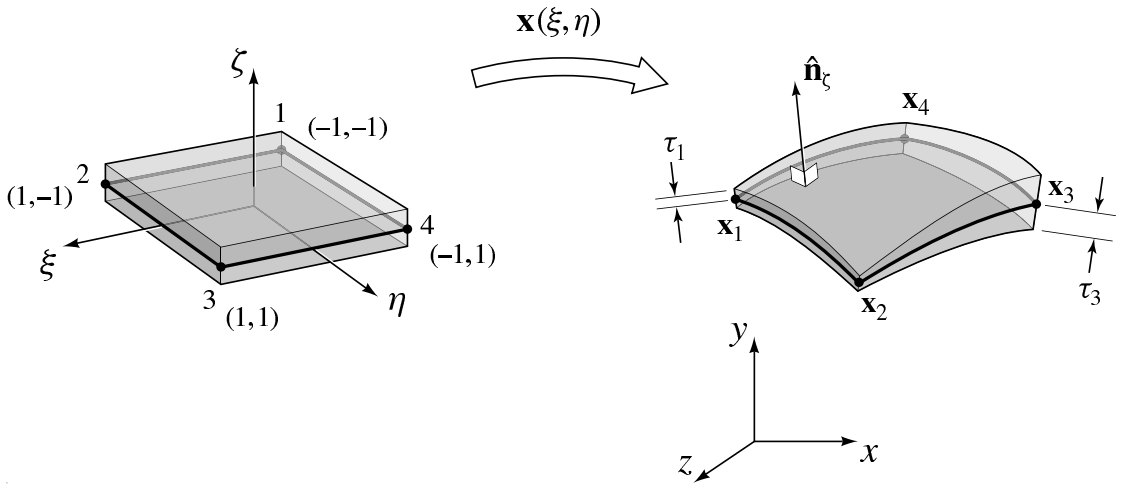


Fig. 3.16: The midplane of the master quadrilateral shell element is mapped to its image in physical coordinates (x, y, z), and the variation in the shell thickness is represented with the element shape functions.

Currently, Aria only supports constant thickness shells, and the simplifications discussed in *3D Triangular Shell Element* also apply in this case. The volume integral therefore reduces to the product of the shell thickness and the integral over the mid-plane surface, with the transformation from computational to physical coordinates identical to that which is described in *2D Quadrilateral Element*.

3.10 CVFEM

As an alternative to the Galerkin Finite Element method (FEM), Aria offers a Control Volume Finite Element (CVFEM) discretization. CVFEM is a general class of numerical methods which can be applied to many problem types, though they are most commonly applied to incompressible Navier-Stokes equations of fluid mechanics. Use of CVFEM in Aria is not limited to fluid mechanics, so this section provides a brief overview of the method applied to an arbitrary conservation equation.

Note: This write-up provides a high-level overview of the CVFEM approach. The reader is referred to [29, 30] or Chapter 7 of [31] for a more detailed description of the approach employed here.

The standard mesh configuration for vertex-centered CVFEM has all DOFs collocated at the grid points, also called nodes. The nodes are the vertices of the finite-elements, as shown in Fig. 3.17. The finite-volumes, also called control volumes, are centered about the nodes. Each element contains a set of sub-faces that define control-volume surfaces. The sub-faces consist of the segments or surfaces that bisect the element faces.

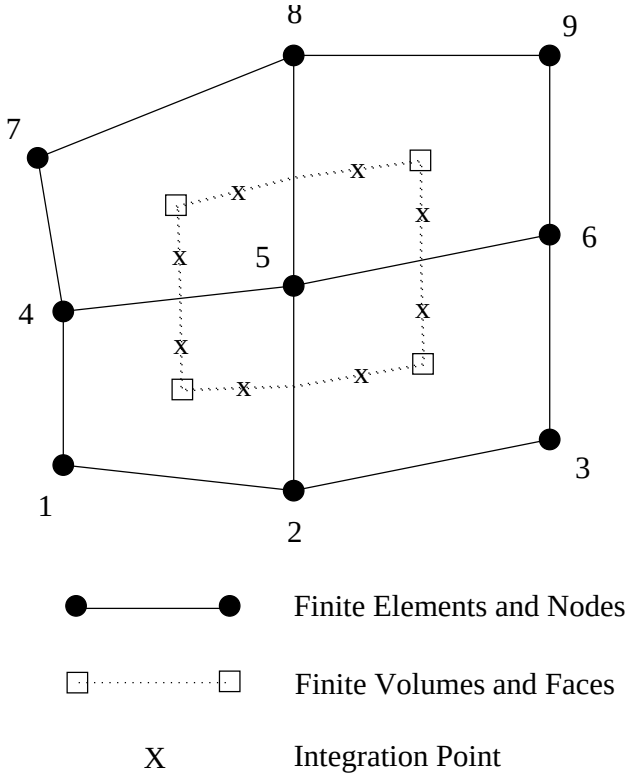


Fig. 3.17: Control Volume is Centered about Finite-Element Node

Integral Form

We begin with the *transient weak description* (or equivalently the *stationary description*). As is discussed in [31], the test function is chosen to be a Heaviside function – unity within the control volume and zero outside of it. The gradient of this test function is zero everywhere but the boundaries of the control volume, where it is given by a negative Dirac delta function. This implies that

$$-\int_{\Omega} \nabla w \cdot \boldsymbol{f} = \sum_{e} \int_{\Gamma_e} \boldsymbol{f} \cdot d\boldsymbol{\Gamma}_{\boldsymbol{e}} \ ,$$

and the classic finite volume form of the *general conservation equation* is recovered

$$\sum_e \int_{\Omega_e} \left(\frac{\partial \phi}{\partial t} - \phi_V \right) d\Omega_e + \sum_e \int_{\Gamma_e} (\boldsymbol{v}\phi - \boldsymbol{f}) \cdot \boldsymbol{d\Gamma_e} = 0 \tag{3.190}$$

Here the summation is over each control volume boundary, Γ_e, with outward pointing differential area vector $\boldsymbol{d\Gamma_e}$.

Note: This integral form is more commonly derived by integrating the *general conservation equation* over the control volume and applying the Gauss Divergence Theorem to the result.

CVFEM Operators

Control volumes (the mesh dual) are constructed about the nodes, as shown in Fig. 3.17. As is discussed in *Finite Element Approximation*, the finite element basis is used to interpolate quantities from the nodes to the interiors of the elements.

Each element contains a set of subfaces that define subcontrol surfaces (SCS). The subfaces consist of line segments (2-D) or surfaces (3-D). The 2-D segments are connected between the element centroid and the edge centroids. The 3-D surfaces are connected between the element centroid, the element face centroids, and the edge centroids. The fluxes across this surface are *integrated over the SCS element* and the contribution is added to the two connected control volumes.

Integration points also exist within the subcontrol volume (SCV) centroids. Such integration points are used for volume integrals such as source terms, the mass matrix, and gradients as needed. These quantities are similarly numerically *integrated over the SCV element*.

For transient problems, the time variation of the nodal values is treated in the same way as the *Galerkin FEM approach*.

3.11 Solution Strategy

In *Governing Equations*, the residual was defined for each equation supported by Aria. *Discretization* then discussed how the equation was discretized over the discrete domain of the problem. Finally, the present section discusses how the solution is determined. First, we define the solution vector $\boldsymbol{\phi}$ as the collection of conserved quantities for each equation solved on each mesh node. Similarly, the residual vector $\boldsymbol{R}$ is defined as the corresponding residual for the given equation and mesh node. The goal of this section is to find the solution vector $\boldsymbol{\phi}$ which satisfies

$$\boldsymbol{R}(\boldsymbol{\phi}, t) = \boldsymbol{0} \tag{3.191}$$

to an acceptable tolerance. We first define the Jacobian as the tangent sensitivity of the residual to the solution vector at the point $\boldsymbol{\phi}$

$$\boldsymbol{J}_{ij} = \left.\frac{\partial r_i}{\partial \phi_j}\right|_{\boldsymbol{\phi},t}$$

To solve (3.191), two approaches are possible. First, Newton's method is a fixed point root finding algorithm. For each iteration k, the Jacobian is used to determine the unique root which sets the residual of the linear approximation of the problem to zero

$$\begin{aligned}\boldsymbol{J}(\boldsymbol{\phi}^k, t^k)\Delta\boldsymbol{\phi}^k &= -\boldsymbol{R}(\boldsymbol{\phi}^k, t^k)\\ \boldsymbol{\phi}^{k+1} &= \boldsymbol{\phi}^k + \alpha\Delta\boldsymbol{\phi}^k\end{aligned}$$

Since the problem is typically nonlinear, this linear approximation will not correspond to the exact root of the real problem. Instead, the iteration is repeated with each producing a better approximation of the exact solution. The algorithm is terminated when a sufficiently close approximation is found.

For the standard Newton's method, $\alpha = 1$. For non-convex problems, though, Newton's method can struggle to converge. In these cases, the second option adds a backtracking line search along the Newton direction $\Delta\boldsymbol{\phi}^k$. This approach finds an appropriate under-relaxation factor α to ensure progress in reducing the problem's residual. For each sub-iteration, the relaxation is reduced by a factor τ. The relaxation adjustment is stopped when the Armijo-Goldstein condition is met

$$\|\boldsymbol{R}(\boldsymbol{\phi}^k + \alpha\Delta\boldsymbol{\phi}^k)\| \leq (1 - \tau\alpha)\,\|\boldsymbol{R}(\boldsymbol{\phi}^k + \Delta\boldsymbol{\phi}^k)\|$$

For a steady problem, the nonlinear system solve must be evaluated only once. Alg. 3.11 below demonstrates the solve of a single nonlinear system

The transient problem can be viewed simply as a series of nonlinear solves, where the transient information is embedded in the residuals. The transient implementation of this algorithm (Alg. 3.11) is provided below, where each timestep solves the nonlinear system at the new time with Alg. 3.11. The initial guess for each step is generated with a prediction of the new solution given its current rate of change, with Alg. 3.11 used to correct this prediction. *Predictor-Corrector* below discusses more details on this strategy. A discussion on how the transient information is included in the system of equations can be found in *Time Integration*.

In more complicated problems, it is common to split the full set of equations into multiple *equation systems* (See *simulation setup* for more discussion on how a segregated solve is setup). In this case, each step must achieve convergence of all segregated equations systems. Subject to the *user specified convergence criteria*, Alg. 3.11 is applied to each equation system and the solution is only accepted when the problem is globally converged

Algorithm 1 Strategy for solving the coupled nonlinear system

Input: $\boldsymbol{\phi}^*, t, \epsilon, I$
Output: $\boldsymbol{\phi}$

Pre-step work (contact search, initialization, etc)
Assemble $\boldsymbol{R}^* = \boldsymbol{R}(\boldsymbol{\phi}^*, t)$
$i = 0$
$\boldsymbol{\phi} = \boldsymbol{\phi}^*$
while $(\|\boldsymbol{R}^*\| > \epsilon)$ **and** $(i < I)$ **do**
    Assemble $\boldsymbol{J}^* = \boldsymbol{J}(\boldsymbol{\phi}^*, t)$
    Solve $\boldsymbol{J}^* \Delta\boldsymbol{\phi} = -\boldsymbol{R}^*$ for $\Delta\boldsymbol{\phi}$
    $\alpha = 1$
    $r_p = \|\boldsymbol{R}^*\|$
    while not finished **do**
        $\boldsymbol{\phi}^* = \boldsymbol{\phi} + \alpha\Delta\boldsymbol{\phi}$
        Assemble $\boldsymbol{R}^* = \boldsymbol{R}(\boldsymbol{\phi}^*, t)$
        if line search not enabled **or** $\|\boldsymbol{R}^*\| < (1 - \tau\alpha) r_p$ **then**
            finished = **true**
        else
            $\alpha = \tau\alpha$
        end if
    end while
    Increment i
    $\boldsymbol{\phi} = \boldsymbol{\phi}^*$
end while
Post-step work (postprocessors, element death, etc)

Algorithm 2 Nonlinear solution strategy for single equation-system transient problem

Input: $\boldsymbol{\phi}_0, \Delta t_0, t_0, t_{\text{end}}, \epsilon, I$
Output: $\boldsymbol{\phi}$

$t = t_0$
$\boldsymbol{\phi} = \boldsymbol{\phi}_0$
$\Delta t = \Delta t_0$
while $t < t_{\text{end}}$ **do**
    Calculate the predicted solution $t \rightarrow t + \Delta t$, $\boldsymbol{\phi} \rightarrow \boldsymbol{\phi}^*$
    Solve coupled system at time $t + \Delta t$ for $\boldsymbol{\phi}$ with initial guess $\boldsymbol{\phi}^*$
    $t = t + \Delta t$
    Calculate new Δt
end while

Algorithm 3 Segregated nonlinear solution strategy for transient problem

Input: $\boldsymbol{\phi}_0, \Delta t_0, t_0, t_{\text{end}}, \epsilon, I$
Output: $\boldsymbol{\phi}$

$t = t_0$
$\boldsymbol{\phi} = \boldsymbol{\phi}_0$ for each equation system
$\Delta t = \Delta t_0$
while $t < t_{\text{end}}$ **do**
  while not globally converged **do**
    for each equation system **do**
      if is first iteration **then**
        Calculate the predicted solution $t \rightarrow t + \Delta t$, $\boldsymbol{\phi} \rightarrow \boldsymbol{\phi}^*$
      else
        Reuse previous solution $\boldsymbol{\phi}^* = \boldsymbol{\phi}$
      end if
      Solve coupled system at time $t + \Delta t$ for $\boldsymbol{\phi}$ with initial guess $\boldsymbol{\phi}^*$
    end for
  end while
  $t = t + \Delta t$
  Negotiate new Δt across all equation systems
end while

Time Integration

As was mentioned above, time information is embedded in the forcing vector of the problem via the `MASS` term (See *Equation Specification* for more information on specifying a transient equation). When this is active, assembly of the `MASS` term considers not only the new time state $n + 1$, but also the respective older states required by the time integration method. Based on the *method* chosen, the rate of change of the solution vector at the new time is given below

`BACKWARD_EULER`, `FIRST_ORDER`
: Backward Euler method is an implicit method where the solution is progressed using only the old and new solution vector. It is also known as BDF1.

$$\dot{\boldsymbol{\phi}}_{n+1} = \frac{\boldsymbol{\phi}_{n+1} - \boldsymbol{\phi}_n}{\Delta t_n}$$

`BDF2`
: Similar to `BACKWARD_EULER`, but uses an additional old time state to achieve 2nd order accuracy.

$$\dot{\boldsymbol{\phi}}_{n+1} = c_{n+1}\,\boldsymbol{\phi}_{n+1} + c_n\,\boldsymbol{\phi}_n + c_{n-1}\,\boldsymbol{\phi}_{n-1}$$

where

$$\gamma = \frac{\Delta t_n}{\Delta t_{n-1}}$$
$$c_{n+1} = \frac{1+2\gamma}{(1+\gamma)\Delta t_n}$$
$$c_n = -\frac{1+\gamma}{\Delta t_n}$$
$$c_{n-1} = \frac{\gamma^2}{(1+\gamma)\Delta t_n}$$

MIDPOINT_RULE, SECOND_ORDER

This method achieves 2nd order accuracy using the old and new solution vector as well as the old rate of change. This method is also known as Crank-Nicolson.

$$\dot{\boldsymbol{\phi}}_{n+1} = \frac{2\left(\boldsymbol{\phi}_{n+1} - \boldsymbol{\phi}_n\right)}{\Delta t_n} - \dot{\boldsymbol{\phi}}_n$$

AVERAGE_ACCELERATION

This method uses the extended mean value theorem to find the averaged acceleration, and is also known as the Newmark-beta method

$$\dot{\boldsymbol{\phi}}_{n+1} = \frac{\boldsymbol{\phi}_{n+1} - \boldsymbol{\phi}_n}{\gamma \Delta t_n} - \frac{1-\gamma}{\gamma}\dot{\boldsymbol{\phi}}_n$$

with $\gamma = 0.5$ and $\beta = 0.25$.

Predictor-Corrector

In order to decrease the number of iterations required in the nonlinear solve, as well as to avoid unphysical initial guesses, the transient iteration begins with a predicted solution. The predictor defaults to the same order as the chosen *Time Integration* method, or can be *set manually*. The first order prediction of the new solution is given by

$$\boldsymbol{\phi}^*_{n+1} = \boldsymbol{\phi}_n + \Delta t_n \, \dot{\boldsymbol{\phi}}_n$$

Similarly, the second order prediction of the new solution is

$$\boldsymbol{\phi}^*_{n+1} = \boldsymbol{\phi}_n + \frac{\Delta t_n}{2}\left[\left(2 + \frac{\Delta t_n}{\Delta t_{n-1}}\right)\dot{\boldsymbol{\phi}}_n - \left(\frac{\Delta t_n}{\Delta t_{n-1}}\right)\dot{\boldsymbol{\phi}}_{n-1}\right]$$

3.12 Coupling

In complex simulation, features may be required that are outside the capabilities of Aria. In these cases, coupling to another application is required to combine functionality. Examples of this could include

- A thermo-mechanical simulation, where displacement is solved by Adagio while temperature is solved by Aria.
- A reentry vehicle in flight, where the fluid domain is solved by Sparc while the solid domain is solved by Aria.

The following sections discuss the concept and implications behind such coupling approaches

One Way Coupling

Assume some "leader" application has state vector $\boldsymbol{\phi}_1$, and its "follower" application has state vector $\boldsymbol{\phi}_2$. In a one way coupling, the solution of the leader app is not a function of the follower app (or its influence can be neglected or approximated offline). In this case, the system of equations is

$$\frac{\partial \boldsymbol{\phi}_1}{\partial t} = F_1\left(\boldsymbol{\phi}_1, t\right)$$
$$\frac{\partial \boldsymbol{\phi}_2}{\partial t} = F_2\left(\boldsymbol{\phi}_1, \boldsymbol{\phi}_2, t\right)$$

Because the leader does not depend on the follower, one needs only to run the leader first and record output over the simulation time. By interpolating these recorded results over time in the follower simulation, both results can be obtained. With some interpolant $\boldsymbol{\phi}_1'(t)$ acting on these recorded results, the system of the follower takes the form

$$\frac{\partial \boldsymbol{\phi}_2^{n+1}}{\partial t} = F_2\left(\boldsymbol{\phi}_1'(t^{n+1}), \boldsymbol{\phi}_2^{n+1}, t^{n+1}\right)$$

The benefit of this approach is that applications can run on their own timescales, and need not adhere to the same MPMD coupling pattern. See *File Based Coupling* for more information on how one can setup a case like this.

Two Way Coupling

Consider now a leader/follower system where their equations now implicitly depend on one another over some coupled subdomain Ω_c. Unlike the *One Way Coupling*, the solutions to both problems must now be advanced together

$$\frac{\partial \boldsymbol{\phi}_1}{\partial t} = F_1\left(\boldsymbol{\phi}_1, \boldsymbol{\phi}_2|_{\Omega_c}, t\right)$$
$$\frac{\partial \boldsymbol{\phi}_2}{\partial t} = F_2\left(\boldsymbol{\phi}_1|_{\Omega_c}, \boldsymbol{\phi}_2, t\right)$$

In a two way coupled simulation, the leader first takes a step with the lagged values of the follower

$$\frac{\partial \boldsymbol{\phi}_1^{n+1}}{\partial t} = F_1\left(\boldsymbol{\phi}_1^{n+1}, \boldsymbol{\phi}_2^{n}|_{\Omega_c}, t^{n+1}\right)$$

This updated $\boldsymbol{\phi}_1$ information is then communicated to the follower, which can then use the new solution to update its own state

$$\frac{\partial \boldsymbol{\phi}_2^{n+1}}{\partial t} = F_2\left(\boldsymbol{\phi}_1^{n+1}|_{\Omega_c}, \boldsymbol{\phi}_2^{n+1}, t^{n+1}\right)$$

In practice, the communication phase is implemented in differing forms depending on the coupling type:

- For coupling to Sierra apps (e.g. Fuego or Adagio) one would use *Sierra Coupling*.
- For external apps like Sparc on the other hand, one would use *MPMD Coupling*.

Algorithm 4 Explicit strategy for two way coupling

1: Leader & Follower initialize $\boldsymbol{\phi}_1^0/\boldsymbol{\phi}_2^0$
2: Follower sends $\boldsymbol{\phi}_2^0|_{\Omega_c}$ to Leader
3: **while** $t < t_{end}$ **do**
4: Negotiate Δt^{n+1}
5: Leader updates $\boldsymbol{\phi}_1^{n+1}$ given $F_1\left(\boldsymbol{\phi}_1^{n+1}, \boldsymbol{\phi}_2^{n}, t^{n+1}\right)$
6: Leader sends $\boldsymbol{\phi}_1^{n+1}|_{\Omega_c}$ to Follower
7: Follower updates $\boldsymbol{\phi}_2^{n+1}$ given $F_2\left(\boldsymbol{\phi}_1^{n+1}, \boldsymbol{\phi}_2^{n+1}, t^{n+1}\right)$
8: Follower sends $\boldsymbol{\phi}_2^{n+1}|_{\Omega_c}$ to Leader
9: Print results
10: **end while**

See *Solution Strategy* for more details on how Aria updates its solutions.

Implicit

This explicit coupling strategy can lead to convergence issues, particularly if the coupling between leader and follower is strong or if the coupling quantities significantly influence the behavior of each system. Since the systems are solved sequentially, there is a lag in the response of each system to changes in the other, restricting the accuracy of the approach. It can also lead to numerical instability, as perturbation in the solution could cause further divergence from the true solution. To address potential convergence issues, an implicit coupling approach can be used instead. In this case, the update step is repeated until the solutions of both apps stop changing. While this increases the computation cost, it ensure convergence in strongly coupled systems and can often allow for larger step sizes.

Convergence is determined based on the change in the solution vector between successive implicit iterations. The simulation is considered to have converged when this difference falls below a given

Algorithm 5 Implicit strategy for two way coupling

Leader & Follower initialize $\boldsymbol{\phi}_1^0/\boldsymbol{\phi}_2^0$
Follower sends $\boldsymbol{\phi}_2^0|_{\Omega_c}$ to Leader
while $t < t_{end}$ **do**
  Negotiate Δt^{n+1}
  Initialize $\boldsymbol{\phi}_i^*$
  while Check Convergence with $F_1\left(\boldsymbol{\phi}_1, \boldsymbol{\phi}_1^*\right)$ and $F_2\left(\boldsymbol{\phi}_2, \boldsymbol{\phi}_2^*\right)$ **do**
    Leader updates $\boldsymbol{\phi}_1^{n+1}$ given $F_1\left(\boldsymbol{\phi}_1^{n+1}, \boldsymbol{\phi}_2^*, t^{n+1}\right)$
    Leader sends $\boldsymbol{\phi}_1^{n+1}|_{\Omega_c}$ to Follower
    Follower updates $\boldsymbol{\phi}_2^{n+1}$ given $F_2\left(\boldsymbol{\phi}_1^*, \boldsymbol{\phi}_2^{n+1}, t^{n+1}\right)$
    Follower sends $\boldsymbol{\phi}_2^{n+1}|_{\Omega_c}$ to Leader
    Record $\boldsymbol{\phi}_i \rightarrow \boldsymbol{\phi}_i^*$
  end while
  Print results
end while

tolerance for both applications, indicating that subsequent iterations yield only negligible changes in the solution vector. This tolerance is chosen based on the desired accuracy of the simulation and the magnitude of the DOFs, and can be given either as an absolute or relative tolerance. Aria supports *implicit tolerances* ϵ_N for a set of different order N norms, which return converged when

$$\frac{|\boldsymbol{\phi}_i - \boldsymbol{\phi}_i^*|_N}{S} \leq \epsilon_N$$

where S is defined as

No Scaling
$S = 1$

DOF_AMAX
$S = |\boldsymbol{\phi}_i|_\infty$

DOF_RMS
$S = \sqrt{\left(|\boldsymbol{\phi}_i|_2\right)^2 / n_{\text{nodes}}}$

Subcycling

Depending on the phenomena being modeled, there could be a difference in timescales between the two coupled apps. An explicit/implicit coupling would require that both apps step at the smaller of the two timescales, leading to unnecessary work for the app with the larger timescales. To overcome this, subcycling can be used. In this case, one app is permitted to take multiple smaller steps to reach the desired "sync step" when results are synced between the two apps. This can significantly alleviate computational load, at the expense of reducing the feedback between apps.

Algorithm 6 Subcycling strategy for two way coupling

Leader & Follower initialize $\boldsymbol{\phi}_1^0/\boldsymbol{\phi}_2^0$
Follower sends $\boldsymbol{\phi}_2^0|_{\Omega_c}$ to Leader
while $t < t_{end}$ **do**
  Negotiate sync step Δt^{n+1}
  for Each Leader substep (or single step if not subcycling) **do**
    Leader calcutes its own Δt^*
    Leader updates $\boldsymbol{\phi}_1^*$ given $F_1\left(\boldsymbol{\phi}_1^*, \boldsymbol{\phi}_2^n, t_1^*\right)$
  end for
  Leader records $\boldsymbol{\phi}_1^* \rightarrow \boldsymbol{\phi}_1^{n+1}$
  Leader sends $\boldsymbol{\phi}_1^{n+1}|_{\Omega_c}$ to Follower
  for Each Follower substep (or single step if not subcycling) **do**
    Follower updates $\boldsymbol{\phi}_2^*$ given $F_2\left(\boldsymbol{\phi}_1^{n+1}, \boldsymbol{\phi}_2^*, t_2^*\right)$
  end for
  Follower records $\boldsymbol{\phi}_2^* \rightarrow \boldsymbol{\phi}_2^{n+1}$
  Follower sends $\boldsymbol{\phi}_2^{n+1}|_{\Omega_c}$ to Leader
  Print results
end while

This is accomplished using either *Solution Control* for a *Sierra Coupling*, or setting the *MPMD timestep mode* for an *MPMD Coupling*.

3.13 Arbitrary Lagrangian-Eulerian

Aria supports solving general transport equations on grids with arbitrary grid motion using an Arbitrary Lagrangian-Eulerian formulation. The following is a brief introduction based on a series of detailed lecture notes [32].

Consider a region undergoing arbitrary mesh motion described by the mapping $\mathbf{x} = \hat{\Phi}(\chi, t)$, where $\mathbf{x}$ is a spatial coordinate and χ is the referential coordinate. For this region, conservation of mass written with respect to the current spatial configuration is

$$\int_{\Omega_x = \hat{\Phi}(\Omega_\chi)} \hat{J}^{-1} \left.\frac{\partial \hat{J}\rho}{\partial t}\right|_\chi + \nabla \cdot (\rho \mathbf{c}) = 0.$$

Here, ρ is the density, $\mathbf{c} = \mathbf{v} - \mathbf{v}_m$ is the convective velocity with $\mathbf{v}$ the material velocity and $\mathbf{v}_m$ the mesh velocity. The time derivative here is with χ held fixed i.e., it is the referential time derivative. The gradient operator ∇, unless otherwise stated, is with respect to the spatial coordinate $\mathbf{x}$. The determinant of the mesh Jacobian $\hat{J}$ is defined as

$$\hat{J} = \det \hat{\mathbf{F}},$$

where the mesh Jacobian is $\hat{\mathbf{F}} = \nabla_\chi \hat{\Phi}$. Using the following relation for referential time derivative

of the determinant of the mesh Jacobian

$$\left.\frac{\partial \hat{J}}{\partial t}\right|_{\chi} = \hat{J}\nabla \cdot \mathbf{v}_m,$$

the strong form, with the advective term written in terms of the divergence of the relative mass flux is

$$\left.\frac{\partial \rho}{\partial t}\right|_{\chi} + \nabla \cdot (\rho \mathbf{c}) + \rho \nabla \cdot \mathbf{v}_m = 0.$$

The equivalent strong form with an advective term written in non-relative mass flux divergence form is

$$\left.\frac{\partial \rho}{\partial t}\right|_{\chi} + \nabla \cdot (\rho \mathbf{v}) - \mathbf{v}_m \cdot \nabla \rho = 0.$$

Note that in either form, the last term that appears is a mesh motion correction term. For the relative mass flux formulation, this term corrects the mass term, whereas for the non-relative form it corrects both the mass term and advective term i.e., boundary mesh fluxes are not explicitly included.

Lastly, a third form is the so called advective form, i.e.,

$$\left.\frac{\partial \rho}{\partial t}\right|_{\chi} + \mathbf{c} \cdot \nabla \rho + \rho \nabla \cdot \mathbf{v} = 0.$$

Given that the equivalent conservation of mass with respect to a material particle is

$$\frac{D\rho}{Dt} = -\rho \nabla \cdot \mathbf{v},$$

we can obtain the following ALE relation between the material time derivative of some quantity (here the density) and the referential time derivative of the quantity

$$\frac{D\rho}{Dt} = \left.\frac{\partial \rho}{\partial t}\right|_{\chi} + \mathbf{c} \cdot \nabla \rho,$$

This ALE relation can be used to derive advective forms for other equations e.g., for the energy equation

$$\rho C_p \frac{DT}{Dt} = \rho C_p \left(\left.\frac{\partial T}{\partial t}\right|_{\chi} + \mathbf{c} \cdot \nabla T \right).$$

Depending on the equation, Aria supports either one or more of these forms.

3.14 Teko Heuristic Multiphysics Preconditioner

Insert some useful notes here about challenge of multiphysics preconditioning.

Convergence of Block Gauss-Seidel

We start our investigation of general multiphysics preconditioning by considering the use of block Gauss-Seidel, as done by Cyr and coworkers [33]. With the blocked linear operator defined as

$$\mathcal{A} = \begin{bmatrix} A_{11} & A_{12} & \cdots & A_{1n_b} \\ A_{21} & A_{22} & \cdots & A_{2n_b} \\ \vdots & \vdots & \ddots & \vdots \\ A_{n_b1} & A_{n_b2} & \cdots & A_{n_bn_b} \end{bmatrix}, \tag{3.192}$$

the corresponding block Gauss-Seidel preconditioner is denoted as

$$\tilde{\mathcal{M}}(\mathcal{A})^{-1} = \begin{bmatrix} A_{11} & A_{12} & \cdots & A_{1n_b} \\ & A_{22} & \cdots & A_{2n_b} \\ & & \ddots & \vdots \\ & & & A_{n_bn_b} \end{bmatrix}^{-1}. \tag{3.193}$$

To analyze the convergence rate of the block Gauss-Seidel preconditioner, let us consider the stationary iterative method

$$\underline{x}^{k+1} = \underline{x}^k + \tilde{\mathcal{M}}(\mathcal{A})^{-1}\left(\underline{b} - \mathcal{A}\underline{x}^k\right). \tag{3.194}$$

The error at each step of this scheme is

$$\underline{e}^{k+1} = \left(I - \tilde{\mathcal{M}}(\mathcal{A})^{-1}\mathcal{A}\right)\underline{e}^k. \tag{3.195}$$

Applying $\|\cdot\|^2_{\tilde{\mathcal{M}}(\mathcal{A})^T\tilde{\mathcal{M}}(\mathcal{A})}$ to both sides of (3.195), we have that

$$\begin{aligned} \|\underline{e}^{k+1}\|^2_{\tilde{\mathcal{M}}(\mathcal{A})^T\tilde{\mathcal{M}}(\mathcal{A})} &= \|\left(I - \tilde{\mathcal{M}}(\mathcal{A})^{-1}\mathcal{A}\right)\underline{e}^k\|^2_{\tilde{\mathcal{M}}(\mathcal{A})^T\tilde{\mathcal{M}}(\mathcal{A})} \\ &= \left(\underline{e}^k\right)^T\left(I - \tilde{\mathcal{M}}(\mathcal{A})^{-1}\mathcal{A}\right)^T\tilde{\mathcal{M}}(\mathcal{A})^T\tilde{\mathcal{M}}(\mathcal{A})\left(I - \tilde{\mathcal{M}}(\mathcal{A})^{-1}\mathcal{A}\right)\underline{e}^k \\ &= \left(\underline{e}^k\right)^T\left(\tilde{\mathcal{M}}(\mathcal{A}) - \mathcal{A}\right)^T\left(\tilde{\mathcal{M}}(\mathcal{A}) - \mathcal{A}\right)\underline{e}^k \\ &\leq \|\tilde{\mathcal{M}}(\mathcal{A}) - \mathcal{A}\|_F^2 \cdot \|\underline{e}^k\|_2^2. \end{aligned}$$

While the bound in (3.196) does not allow for direct comparison between $\underline{e}^k$ and $\underline{e}^{k+1}$ given the difference in norms, it nevertheless demonstrates that minimizing $\|\tilde{\mathcal{M}}(\mathcal{A}) - \mathcal{A}\|_F^2$ improves the convergence of the block Gauss-Seidel approach.

Linear Ordering Problem for Block Gauss-Seidel

We now consider the linear ordering $\mathcal{R}(\cdot)\,C$, a simultaneous permutation of rows and columns, from which to construct $\tilde{\mathcal{M}}(\mathcal{RAC})$ that reduces the relative error reduction bound in (3.196) when replacing $\mathcal{A}$ with $\mathcal{RAC}$ and $\tilde{\mathcal{M}}(\mathcal{A})$ with $\tilde{\mathcal{M}}(\mathcal{RAC})$. Note that the cost of applying $\mathcal{RAC}$ and $\tilde{\mathcal{M}}(\mathcal{RAC})$ does not depend on the ordering. As such, the convergence of the block Gauss-Seidel preconditioner can be improved at no additional expense by optimizing the error bound in (3.196). Substituting the $\mathcal{RAC}$ and $\tilde{\mathcal{M}}(\mathcal{RAC})$ into the right hand side of (3.196) leads to an optimization problem of the form: *find block numbering permutation* $\mathcal{R}^*(\cdot)\,C^*$ *such that*

$$(\mathcal{R}^*,\ C^*) = \text{argmin}_{\forall \mathcal{R},\ C}\ \|\tilde{\mathcal{M}}(\mathcal{RAC}) - \mathcal{RAC}\|_F^2. \tag{3.196}$$

By constructing a $(n_b,\ n_b)$ matrix H with $H_{ij} = \|\mathcal{A}_{ij}\|_F^2$, the discrete optimization problem in (3.196) is equivalent to the matrix triangulation problem: *Determine a simultaneous permutation of rows and columns of a matrix that maximizes the sum of the superdiagonal entries* [34]. The matrix triangulation problem is further equivalent to the linear ordering problem, which is NP-hard with worst-case $\Omega(n_b!)$ complexity.

While the linear ordering problem is NP-hard, this problem is well studied with good exact and heuristic methods (see [34] and references therein). We employ the exact method as formulated by Charon and Hudry [35], which uses a branch-and-bound approach with Lagrangian relaxation. We denote solving the linear ordering problem in (3.196) using this scheme as `lop`.

Block Merging Problem for Block Gauss-Seidel

While the optimization problem in (3.196) can be used to determine the optimal ordering, the convergence of the block Gauss-Seidel scheme may be hampered when two physics are strongly coupled in both directions. In this way, it helps to treat such blocks *monolithically*, rather than relying on the one-way coupling encapsulated in the block Gauss-Seidel preconditioner. For example,

$$\underbrace{\begin{bmatrix}\begin{bmatrix}A & B\\ D & E\end{bmatrix} & \begin{bmatrix}C\\ F\end{bmatrix}\\ & J\end{bmatrix}^{-1}}_{\tilde{\mathcal{M}}(\mathcal{A}')^{-1}}\underbrace{\begin{bmatrix}\begin{bmatrix}A & B\\ D & E\\ G & H\end{bmatrix} & \begin{bmatrix}C\\ F\\ J\end{bmatrix}\end{bmatrix}}_{\mathcal{A}'} \tag{3.197}$$

may be much better conditioned than

$$\underbrace{\begin{bmatrix}A & B & C\\ & E & F\\ & & J\end{bmatrix}^{-1}}_{\tilde{\mathcal{M}}(\mathcal{A})^{-1}}\underbrace{\begin{bmatrix}A & B & C\\ D & E & F\\ G & H & J\end{bmatrix}}_{\mathcal{A}}, \tag{3.198}$$

assuming that the off-diagonal physics coupling B and E are both large. We denote that a block merging operation is one that treats a two by two sub-block system as a single monolithic

sub-block. Similar to the block reordering problem, an optimization problem based on the error bound in (3.196) may be posed: *find the minimum number of block merging operations such that*

$$\|\tilde{\mathcal{M}}\left(\mathcal{A}'\right) - \mathcal{A}'\|_F \leq \tau, \tag{3.199}$$

where τ represents a user-defined target error reduction in (3.196). While the optimization problem posed in (3.199) does not enjoy the same parallel to the linear ordering problem as does the optimization problem in (3.196), we may nevertheless employ a *greedy* block merging approach that utilizes the linear ordering problem solver. This is demonstrated in Alg. 3.14, which relies on the linear ordering problem solver `lop` as described in *Linear Ordering Problem for Block Gauss-Seidel*.

Algorithm 7 Greedy block merging algorithm

Input: Matrix $\mathcal{A}$ with n_b blocks, user-provided threshold τ.
Output: Re-grouped matrix $\mathcal{A}'$ with $n_b' < n_b$ blocks.

$\mathcal{A}^{(0)} \leftarrow \texttt{lop}(\mathcal{A})$
$k \leftarrow 0$
while $\|\tilde{\mathcal{M}}\left(\mathcal{A}^{(k)}\right) - \mathcal{A}^{(k)}\|_F \geq \tau$ **do**
  $k \leftarrow k + 1$
  $(i^*, j^*) \leftarrow \text{argmax}_{i,j,i>j} \|\mathcal{A}_{ij}^{(k-1)}\|_F$
  $\mathcal{A}^{(k)} \leftarrow \texttt{lop}(\texttt{combineBlocks}(\mathcal{A}^{(k-1)}, i^*, j^*))$
end while
return $\mathcal{A}^{(k)}$

Choosing Sub-block Solvers for Block Gauss-Seidel

The block Gauss-Seidel preconditioner defined in (3.193) assumes that the sub-block inverses required along the diagonal are exact. While the sub-block solvers required are now restricted to relatively easy problems involving one or a few multi-physics interactions, the performance of the block preconditioning strategy is highly dependent on being able to choose good sub-block solvers. We consider the impact of replacing the exact sub-block solvers in the block Gauss-Seidel operator defined in (3.193) with approximate sub-block solvers,

$$\mathcal{M}\left(\mathcal{A}\right)^{-1} = \begin{bmatrix} M_{11} & A_{12} & \cdots & A_{1n_b} \\ & M_{22} & \cdots & A_{2n_b} \\ & & \ddots & \vdots \\ & & & M_{n_bn_b} \end{bmatrix}^{-1} \tag{3.200}$$

where each of the A_{ii} entries from (3.193) are replaced with M_{ii}. The effect in (3.200) is that applying $\mathcal{M}\left(\mathcal{A}\right)^{-1}$ now requires applying M_{ii}^{-1} for each diagonal sub-block, where M_{ii}^{-1} represents a preconditioner or solver approximating the action of A_{ii}^{-1}. A natural question follows: *how does*

the convergence of $\mathcal{M}(\mathcal{A})^{-1}$ *compare to* $\tilde{\mathcal{M}}(\mathcal{A})^{-1}$ *?* This question is addressed in (3.201).

$$\begin{aligned}\kappa\left(\mathcal{M}^{-1}(\mathcal{A})\,\mathcal{A}\right) &\leq \kappa\left(\mathcal{M}^{-1}(\mathcal{A})\,\tilde{\mathcal{M}}(\mathcal{A})\right)\cdot\kappa\left(\tilde{\mathcal{M}}^{-1}(\mathcal{A})\,\mathcal{A}\right)\\ &\geq \underbrace{\max_{\forall M_{ii}^{-1}A_{ii}}\left(\kappa\left(M_{ii}^{-1}A_{ii}\right)\right)}_{\text{Sub-block Solver Conditioning}}\cdot\underbrace{\kappa\left(\tilde{\mathcal{M}}^{-1}(\mathcal{A})\,\mathcal{A}\right)}_{\text{Multi-physics Coupling}}\end{aligned}$$

As a consequence of (3.201), a single resistant sub-block solver can derail the convergence progress of the entire solver. For both robustness and user ease-of-use, we propose using an *adaptive* sub-block solver approach that monitors the relative residual reduction at each preconditioner application to inform the choice of sub-block preconditioner or solver. The adaptive sub-block preconditioner or solver transitions to a more robust preconditioner or solver if the relative residual reduction becomes larger than some user-specified tolerance ε,

$$\frac{\|\underline{b}_i - A_{ii}M_{ii}^{-1}\underline{b}_i\|_2}{\|\underline{b}_i\|_2} > \varepsilon. \tag{3.201}$$

The condition in (3.201) is checked with every sub-block preconditioner/solver evaluation to check that a given sub-block is solved to within the user-specified tolerance. If the condition is true, then the sub-block solver switches to the next solver in a user-specified schedule of sub-block solvers to evaluate. This process continues until a convergent sub-block solver is found, with each successive sub-block solver being more robust, albeit more expensive, than the last. For example, the we employ the following schedule for adaptively solving sub-blocks on CPU architectures:

1. GMRES(50) preconditioned with Jacobi
2. GMRES(30) preconditioned with DD(0)-ILU(0)
3. GMRES(30) preconditioned with DD(1)-ILU(1)
4. GMRES(30) preconditioned with DD(2)-ILU(2)
5. Sparse direct solver

On GPU architectures where the relative expense of constructing the DD-ILU based solvers is greater, we allow for additional GMRES iterations:

1. GMRES(100) preconditioned with Jacobi
2. GMRES(50) preconditioned with DD(0)-ILU(0)
3. GMRES(50) preconditioned with DD(1)-ILU(1)
4. GMRES(50) preconditioned with DD(2)-ILU(2)
5. Sparse direct solver

This ensures that sub-blocks that are relatively easy to precondition may be done so with an inexpensive preconditioned GMRES solve using Jacobi or DD(0)-ILU(0). At the same time, sub-blocks requiring a more robust solver can utilize a more robust preconditioned GMRES

iteration as a high-quality approximate to the sub-block inverse. These two schedules work well for a wide range of sub-block solves encountered. However, they are sub-optimal for sub-blocks associated with the continuity equation. For these cases, we employ the following solver schedule on CPU architectures:

1. GMRES(30) preconditioned with MueLu
2. GMRES(30) preconditioned with DD(0)-ILU(0)
3. GMRES(30) preconditioned with DD(1)-ILU(1)
4. GMRES(30) preconditioned with DD(2)-ILU(2)

and on GPU architectures:

1. GMRES(50) preconditioned with MueLu
2. GMRES(50) preconditioned with DD(0)-ILU(0)
3. GMRES(50) preconditioned with DD(1)-ILU(1)
4. GMRES(50) preconditioned with DD(2)-ILU(2)

We note that the specific adaptive sub-block solver settings are subject to change.

4 Simulation Setup

4.1 Input File Basics

Anatomy of an Input Deck

Every Aria simulation requires a text input file to define the problem. The input file defines everything about your problem except the mesh itself. You can prepare and edit this file in any text editor of your choosing. Using the Sierra Editor in SAW to prepare this file will provide additional context hints and syntax checking in real-time.

Indentation & Capitalization

Indentation is recommended but not required in an Aria input file. You can indent with tabs or spaces, or choose not to indent things at all - although we highly recommend using indentation (with spaces) to make a consistently readable input file and to avoid errors with commands going in the wrong scope.

Aria commands are generally not case sensitive, although commands referring to a file (e.g. the mesh file or an Aprepro include file) will be case sensitive.

Comments

Anything that starts with a # or a $ will be treated as a comment and ignored. Long lines can be split using \$, like this

```
# This is a comment
$ and so is this

# This is a single line command split on three lines
List of Blocks = block_1 block_2 block_3 \$
                 block_4 block_5 block_6 \$
                 block_7 block_8
```

Structure

The Aria input file is structured using nested blocks with `Begin` and `End` commands surrounding each block. The outermost block must always be the SIERRA block. Commands directly in this block are referred to as "Domain-level" or "Sierra-level" commands. Other commands must go inside inner blocks, such as "Procedure" or "Aria Region", and will be referred to as such. If you put a command in the wrong level, you will get an error when you try to run Aria.

```
Begin SIERRA Aria
  # Domain-level commands:
  # * Aria materials
  # * Finite element model
  # * Linear solvers

  Begin Procedure AriaProcedure
    # Procedure-level commands

    Begin Solution Control Description
      # Solution control commands
    End

    Begin Aria Region myRegion
      # Region-level commands:
      # * Equation(s) to solve
      # * Initial and Boundary conditions
      # * Post-processors
      # * File output specifications
    End
  End
End
```

There are two types of commands that can go in the input file - block commands and line commands. Block commands start with the `Begin` keyword and end with the `End` keyword, and can contain line commands or nested blocks.

```
Begin My Block Somename
   # line commands
End My Block Somename
```

The block type and name are optional on the `End` command, but may improve readability for large blocks

```
Begin My Block Somename
   # line commands
End
```

A line command is a command on a single line

```
Density = Constant value = 1.0
```

Each line command has a specific block or set of blocks where it is valid. Using it outside those blocks will result in an error about an un-recognized command.

For simple problems, the governing equations to solve are usually specified in the `Aria Region` block. When using this mode, a single linear system will be constructed with all the equations in the region and solved monolithically. Alternately, you can use `Equation System` blocks within the Aria region to separate out different equations.

```
Begin Aria region myRegionName
  # general region commands

  Begin Equation System myPhysics1
    # physics1 equation
  End

  Begin Equation System myPhysics2
    # physics2 equations
  End
End
```

When using this mode, a linear system will be constructed and solved independently for each equation system. These will be solved in the order listed in the input file.

Syntax checking

It is common that users may misspell or misplace a certain command line/block, causing an error before a run even starts. To streamline this process, Sierra offers a command line option to check the syntactic correctness of an input deck without loading reading the mesh. This can be done for a sample input deck `input.i`:

```
module load sierra
aria -i input.i --check-syntax
```

Typically, this will run quickly and determine whether or not `input.i` has any syntactic errors reported in the log file.

It is important to note that the speed of `--check-syntax` scales with the size of the input file. For example, the following input deck commands

```
Begin Aria region myRegionName
  # general region commands

  Begin Equation System myPhysics1
    Eq energy for temperature on block_1 using Q1 with lumped_mass diff␣
↪src
    Eq energy for temperature on block_2 using Q1 with lumped_mass diff␣
↪src
    #... up to 1000 blocks
    Eq energy for temperature on block_1000 using Q1 with lumped_mass␣
↪diff src
  End
End
```

will run `--check-syntax` much slower than the following

```
Begin Aria region myRegionName
  # general region commands
  mesh group solids = block_1 block_2 ... block_1000 # assume all 1000␣
↪blocks are explicitly defined in this mesh group
  Begin Equation System myPhysics1
    eq energy for temperature on solids using Q1 with lumped_mass diff src
  End
End
```

This is due to having multiple ordering constraints on various commands lines/blocks that Sierra internally resolves before checking the input deck syntax.

Aprepro Scripting

Aria input files can use Aprepro commands to do pre-processing, calculations, and variable management within the input file itself. If you have an input file with Aprepro commands, you can choose to manually run Aprepro on that file to generate a processed output file like this

```
aprepro myfile.in > myfile.out
```

However, this is not necessary. When running Aria it will automatically run Aprepro on the input before parsing it, and the input file contents put in the header of the log file will be the Aprepro-resolved contents.

This section covers basic Aprepro usage as is relevant to general Aria use. The complete Aprepro manual can be found with the SEACAS Documentation.

Defining Constants

An Aprepro constant can be declared inside {} anywhere in the input file. Aprepro constants cannot be re-defined later unless they are marked as mutable by using a leading underscore in their name. Aprepro constants can also be defined from the command line when calling Aria. The values supplied at the command line will override values in the input file. Multiple values should be separated with commas or spaces.

```
aria -i myfile.i --define "N=4"
aria -i myfile.i --define "N=4 M=2"
aria -i myfile.i --define "N=4,M=2"
```

When the input file is processed by Aprepro, the contents of the {} sections will be replaced with their values. For example, the following input

```
# Tref = { Tref = 300 } K
IC for Temperature on all_blocks = constant value = {Tref}
BC Flux for Energy on surface_1 = Nat_Conv T_ref={Tref} h=5
```

would evaluate to this (which would be shown in the log file)

```
# Tref = 300 K
IC for Temperature on all_blocks = constant value = 300
BC Flux for Energy on surface_1 = Nat_Conv T_ref=300 h=5
```

Note: While the line `Tref = {Tref = 300} K` may appear redundant in the input, since the term inside the {} is replaced with its actual value, using descriptive labels outside the {} results in a more readable output in the log file.

You can also do math in Aprepro to define functional values, for example to evaluate a correlation to get a convection coefficient you could use Aprepro like this

```
# Tref = { Tref = 300 } K
# D = { D = 0.1 } m
# nu = { nu = 1e-5 } m2/s
# V = { V = 10 } m/s
# Re = { Re = V*D/nu }
# Pr = { Pr = 0.7 }
# k = { k = 0.05 } W/m-K
# Nu = { Nu = 0.332*Re^0.5*Pr^(1/3) }
# h = { h = Nu*k/D } W/m2-K

BC Flux for Energy on surface_1 = Nat_Conv T_ref={Tref} h={h}
```

Including Files

Aprepro can also be used to separate your input into multiple files, which can help with readability and maintainability of your problem setup. For example, you could have a file with your commonly used Aria material definitions that is shared between multiple input setups. You can also put verbose sections of the input file like material assignment into a separate file to keep the primary file more readable, especially for problems with a large number of blocks. For example:

```
# Material Assignments (materials.inc)
Use material al_6061 for block_1
Use material al_6061 for block_2
Use material al_6061 for block_3
Use material ss_304 for block_4
Use material ss_304 for block_5
Use material ss_304 for block_6
Use material pmdi_foam for block_7
```

```
#{include("parameters.inc")}

Begin Finite Eleement Model foo
    #{include("materials.inc")}
End
```

If-Else Logic

Aprepro allows the use of `if-else` statements, which can be used to conditionally activate parts of your input file. Commands inside a conditional statement will only be included in the actual Aria input if the condition is true.

```
# Use constant flux = { use_const_flux = 1 }
# Do output = { do_output = 0 }

{if(use_const_flux)}
BC Flux for energy on surface_1 = constant value = 1
{else}
BC Flux for Energy on surface_1 = Nat_Conv T_ref=300 h=10
{endif}

{if(do_output)}
Begin Results Output Foo
  # ...
End
{endif}
```

Loops

Aprepro also allows you to define simple loops, which can be helpful when having to repeat very similar commands many times in an input file. For example, if you want to generate an N by M grid of temperature probes you could do

```
# M = { M = 10 }
# N = { N = 15 }
# x0 = { x0 = 0.1 }
# y0 = { y0 = 0.2 }
# dx = { dx = 0.1 }
# dy = { dy = 0.1 }

{NOECHO} { _i = 0 }
{loop(M)}
{NOECHO} {++_i} { _j = 0 }
{loop(N)}
{NOECHO} {++_j}
{ECHO} Postprocess value of expression temperature at location \$
       {x0 + (_i-1)*dx} {y0 + (_j-1)*dy} as Tprobe_{_i}_{_j}
{endloop}
{endloop}
```

Which will generate 150 data probe commands in the input file. You could use similar logic to put those probe variables in the results output or heartbeat output block, which avoids boilerplate and makes it easy to change the probe grid location or resolution. The {NOECHO} and {ECHO} commands turn off and on output to the input file, respectively, and can be used to prevent intermediate output from being printed to the input file.

A similar approach can be used with delimited lists of parameters, such as enclosure surfaces. If you had a list of surfaces, each of which needed an enclosure definition block, you could avoid repeating a nearly identical enclosure block N times using this approach.

```
# enclosure surfaces = { encl_surfs = "surface_1 surface_2 surface_3␣
↪surface_4" }

{NOECHO} { _j = 0 } { num_enclosures = word_count(encl_surfs, " ") }
{loop(num_enclosures)}
{ECHO}# Enclosure { ++_j}
Begin Enclosure Definition enc_{_j}
  Add Surface {get_word(_j, encl_surfs, " ")}
  Blocking Surfaces
  Use Viewfactor Calculation hemicube
  Use Viewfactor Smoothing none
  Use Radiosity Solver rs_gmres
End
{endloop}
```

4.2 Specifying the Mesh

The following section gives a brief overview on how to define a finite element model within Aria from an input finite element mesh. For details on the command usage/reference, the reader is referred to *Finite Element Model Parameters*

Finite element model

The model discretization (mesh database) and the mesh components to be used in the model are defined at the Sierra scope and are later referenced by the application at the Region level. Specification of a mesh database and association of material properties with portions of the mesh are defined within the `Finite Element Model` command block. In coupled analyses the need often arises for different discretizations of the same physical domain. In this case one must supply a different `Finite Element Model` command block for each discretization since only a single mesh can be referenced within this command block.

Defining the Mesh Database and Decomposition Method

The mesh database can be defined as follows:

```
Database name {=|are|is} <StreamName>
```

Where the <StreamName> is the base name containing the output or input mesh. If the filename begins with the / character, it is an absolute path; otherwise the path to the current directory (where the input file is) will be prepended to the name. If this command line is omitted, then an assumed filename is used; this will be the basename of the input file with a .g suffix appended. The most commonly used database formats are Exodus and Genesis (.e and .g suffix, respectively). The database type defines the format used for the input/output mesh:

The mesh can be partitioned across processes for parallel runs through a variety of decomposition methods; the particular method used can be defined with the following command line:

```
Decomposition Method {=|are|is} <MethodName>
```

The most commonly used parameters for <MethodName> are rcb (Recursive Coordinate Geometric Bisection) and rib (Recursive Inertial Bisection). If this command line is not specified, then the default method is rcb if the input mesh file is not already pre-decomposed. A simple example of defining a database name and its decomposition method for a finite element command block is shown below:

```
Begin Finite Element Model The_Model
   Database Name = the_mesh.g
   Decomposition Method = rib
End
```

Coordinate System Specification

The coordinate system is specified with the command

```
Coordinate System {=|are|is} <CoordSystem>
```

where <CoordSystem> is Cartesian, Axisymmetric, or Cylindrical. The Cartesian coordinate system is used for standard 2D and 3D simulations. Axisymmetric is used for scalar-only problems, where the input mesh is 2D and vectors are output with length 2. Cylindrical can be used for both scalar and vector problems, and the vector and tensor output contain all three dimensions. Cylindrical is preferred. For both Cylindrical and Axisymmetric, the symmetry axis is specified with the axisymmetry axis command in the Aria Region block.

Mesh Database Part Names and Assemblies

The input mesh database can be created by a variety of mesh generators. These mesh databases can contain parts of the mesh that either have user-defined names or utilize a default internal naming convention used for the meshed entity/part with a numbered index (e.g. surface_1, block_8). Fig. 4.1 below shows an example of a mesh using the default enumerating scheme.

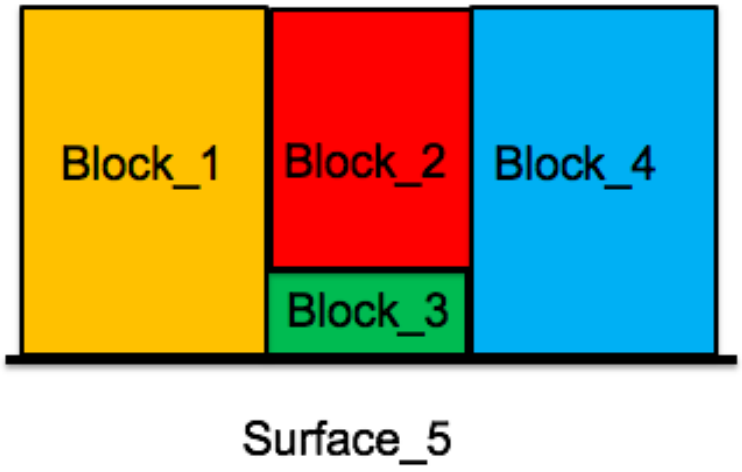


Fig. 4.1: Enumerated blocks

In the following sections, we will refer to Fig. 4.1 to demonstrate example commands of setting up the `finite element model` command block.

Mesh Part Material Assignments

Assignment of the associated material can be accomplished in two different ways. The first way is using a command block

```
Begin Parameters for Block block_2
  material 304L
End

Begin Parameters for Block block_3
  material 304L
End

Begin Parameters for Block block_1
  material cu
End
```

(continues on next page)

(continued from previous page)

```
Begin Parameters for Block block_4
  material cu
End
```

This individually assigns a material 304L to the two inner blocks and cu to the two outer blocks in Fig. 4.1. The above command block can be further shortened by assigning materials to more than one block at a time:

```
Begin Parameters for Block block_2 block_3
  material 304L
End

Begin Parameters for Block block_1 block_4
  material cu
End
```

The other way of assigning materials is an one-line command, which is even more succinct:

```
use material 304L for block_2 block_3
use material cu for block_1 block_4
```

Finite Element Model Parameters shows a summary of commands used to assign materials to parts of the mesh.

Assemblies and Mesh Groups

Assemblies

Assemblies can be defined within Aria to aggregate mesh parts (block, surface or node list) together. An example of defining a surface and block assembly is shown below:

```
Begin assembly flux_bc
  surface = surface_1 surface_2
End assembly

Begin assembly block_test
  block = block_1 block_2
End assembly
```

This block is defined under the finite element command block. It should be noted that each assembly must contain the same type (or rank) of parts (block, surface or node list). Building upon the examples in *Mesh Part Material Assignments*, we can increase the readability of our input deck by doing the following:

```
Begin assembly cu_blocks
  block = block_1 block_4
End assembly

Begin assembly 304L_blocks
  block = block_2 block_3
End assembly

use material 304L for 304L_blocks
use material cu for cu_blocks
```

Assemblies can be used in a variety of operations (i.e. equation definition, boundary conditions and postprocessing operations) and generally help readability of an input file. In addition, these assemblies can be used for transfers to other apps as well. The preferred method for grouping mesh parts together within an input deck is through assemblies. A summary of commands available for defining assemblies is located in *Finite Element Model Parameters*.

Mesh Groups

Another option for aggregating mesh parts together is using the `mesh group` command (which exists at the Aria Region scope):

```
Mesh Group cu_blocks = block_1 block_4
Mesh Group 304L_blocks = block_2 block_3
```

During input processing, mesh groups are updated to assume Assembly attributes. As such, they can be used when transferring variables to other applications. Mesh groups cannot be used directly in other applications, as they are Aria-specific; however mesh groups can group together CDFEM-decomposed parts, whereas assemblies cannot (*Level Set/CDFEM* for more on CDFEM). Mesh groups also have operators that can be applied to mesh parts and refer to other mesh groups. These operators include plus (+), minus (-) and range (`to`, `thru` or `through`) operators. As an example, we can first define a mesh group that holds all of the blocks in Fig. 4.1:

```
Mesh Group summed_volumes = block_1 to block_4
```

The operation `block_1 to block_4` aggregates all of the blocks in Fig. 4.1 into `summed_volumes`. The `304L_blocks` and `cu_blocks` from the example in *Assemblies* can then be defined as:

```
Mesh Group cu_blocks = summed_volumes - block_2 - block_3
Mesh Group 304L_blocks = summed_volumes - block_1 - block_4
```

By default, the following aliases are provided

ALL_BLOCKS/ALL_VOLUMES
Reference to all active element blocks in the mesh

ALL_SURFACES
Reference to all active surfaces in the mesh

ALL_BULK_ELEMENTS
Reference to all active *bulk nodes* in the mesh

Note: The ALL_BLOCKS alias *does not* include the *bulk nodes* in a mesh. To access both the blocks and bulk nodes of a mesh simultaneously, one can use a *mesh group* e.g.

```
MESH GROUP all_blocks_and_bulk_elems = all_blocks all_bulk_
↪elements
```

Omitting Parts of the Mesh

It is possible to omit parts of your input mesh at the beginning of the simulation. Doing so will make it illegal to refer to the omitted volume/block/assembly later in the input file. For example, specifying initial or boundary conditions on the omitted block will be impossible. The entities (elements, faces etc.) unique to the omitted block are never created, as if the omitted parts never existed in the mesh file. However, shared entities (e.g., a surface shared between a real and an omitted volume) will exist and may have conditions (boundary, initial etc.) applied on the portion of the entities supported by the non-omitted volumes. Below shows available commands for omitting parts of the mesh:

```
omit block  <BlockList>
omit volume <VolumeList>
omit assembly <AssemblyList>
```

Details on the syntax usage can be found in *Finite Element Model Parameters*.

Skinning

In Aria, skinning refers to the creation of surface parts making up the exposed boundaries of blocks. In most cases, one wishes to collect the existing faces of these blocks and place them in a new *Skinned Surface* part. In rare cases, one instead wishes to create a *Skinned Shell* with the same geometry as the exposed boundary. These two options are discussed below.

Skinned Surface

The definition of a skinned surface is comprised of 3 main filter types that can be applied to both surfaces and blocks.

`touching surfaces`
: Define the union of surfaces to include in the resulting skinned part. Only faces included in any of the specified surfaces are accepted.

`touching blocks`
: Define the union of blocks' faces to include in the resulting skinned part. Only faces connected to any of the specified blocks are accepted.

`excluding surfaces`
: Define the union of surfaces to exclude in the resulting skinned part. Faces included in any of the specified surfaces are excluded.

`excluding blocks`
: Define the union of blocks' faces to exclude in the resulting skinned part. Faces connected to any of the specified blocks are excluded.

`intersecting surfaces`
: Define the intersection of surfaces to include in the resulting skinned part. Only faces included in all of the specified surfaces are included.

`intersecting blocks`
: Define the intersection of blocks' faces to include in the resulting skinned part. Only faces connected to all of the specified blocks are included.

Note that to ensure the skinned surface definition is self-explanatory, at least one filter line is required. If the entire domain should be skinned, the trivial `touching blocks = all_blocks` filter can be used.

Note: *Assemblies* and hence mesh groups *mesh groups* can both be used to define the extents of skinned surfaces. Although skinned surfaces input is parsed *before* mesh groups, its construction is deferred so that the skinned surface *can* be added to mesh groups.

The default treatment of internal faces with surface skinning varies with use case. Namely, intersection of parts implies that internal sideset(s) are created, so internal faces are included by default when the `intersecting` keyword is used. In all other cases, internal faces are excluded by default so that only the exposed boundary of the surface is selected. If one needs to stray from this default behavior, they can expressly state how internal faces are to be treated with the *internal faces* line.

Examples for some common noteworthy use cases are as follows

Skin exposed boundary of domain or subset of block(s)

A common use case (e.g. when apply convective/rad BCs) is to skin the relevant blocks and apply the BCs to the resulting part. Note that the default behavior here is to exclude internal faces, so clarifying this is optional. An example syntax for this case is as follows

```
# Scope: Sierra > Procedure > Aria Region
begin skinned surface skinned_all
  touching blocks = all_blocks
  # internal faces = exclude # default is to exclude
end
begin skinned surface skinned_b1b2
  touching blocks = block_1 block_2
  # internal faces = exclude # default is to exclude
end
```

The resulting skinned surfaces are demonstrated in Fig. 4.2 for `skinned_all` and `skinned_b1b2` on the left and right respectively.

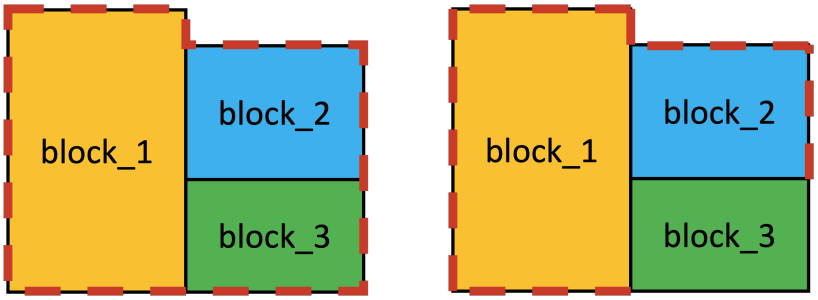


Fig. 4.2: Resulting skinned surface for all blocks in thick dashed red. Left domain shows the results of skinning all blocks, while the right shows the skinning of only `block_1` and `block_2`.

Note that the `exclusion` filter can also be used, meaning that the following would also produce the subset of exposed boundary parts shown in Fig. 4.2

```
# Scope: Sierra > Procedure > Aria Region
begin skinned surface skinned_excluding_b3
  excluding blocks = block_3
end
```

Defining the internal surface at the intersection of blocks

An example of this use case is in defining *interface conditions* between connected block, or *postprocessing* at block interfaces. Here we use the `intersecting` keyword to describe the blocks we want to intersect. Note that with the use of the intersecting keyword, the default behavior is to include internal faces so clarifying this is optional.

```
# Scope: Sierra > Procedure > Aria Region
begin skinned surface interface_b1b2
  intersecting blocks = block_2 block_3
  # internal faces = include # default is to include
end
```

The resulting skinned surface is demonstrated in Fig. 4.3 for `interface_b1b2`

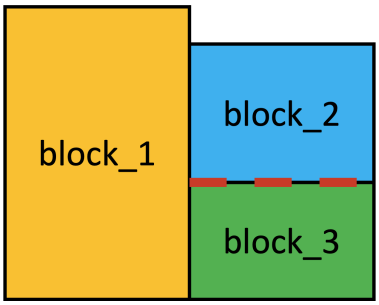


Fig. 4.3: Resulting intersecting skinning surface for `block_2` and `block_3` in thick dashed red

Skinning block(s) and including internal faces

As discussed above, we do not have an `intersecting` line here, so we must manually clarify that internal faces should be included e.g.

```
# Scope: Sierra > Procedure > Aria Region
begin skinned surface skinned_int_ext_block_2
  touching blocks = block_2
  internal faces = include
end
```

The resulting skinned surface is demonstrated in Fig. 4.4 for `skinned_int_ext_block_2`

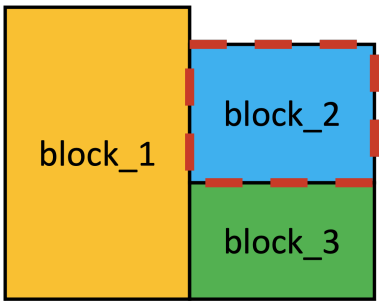


Fig. 4.4: Resulting skinned `block_2` including its internal faces in thick dashed red

Nested skinned parts

Skinned surfaces can also be used within skinned surface definitions. This is useful to define more complex surfaces e.g.

```
# Scope: Sierra > Procedure > Aria Region
Begin Skinned surface to_remove
  intersecting blocks = block_2 block_3
end
Begin Skinned surface skinned_bl_23_ext
  touching blocks = block_2 block_3
  internal faces = include
  excluding surfaces = to_remove
end
```

creates a skinned surface comprised of the exterior of merging `block_2` and `block_3`, as is demonstrated in Fig. 4.5 for `skinned_bl_23_ext`

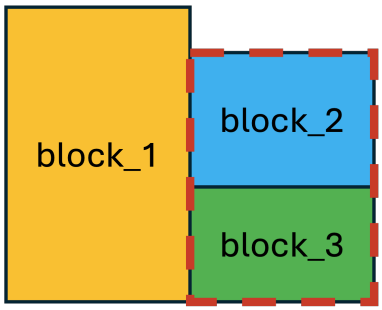


Fig. 4.5: Resulting skinned exterior of `block_2` and `block_3` with shared faced removed

Note: Skinned surfaces are parsed in the order they appear in the input. Ensure that nested surfaces are defined in an appropriate order.

See the *command reference* for more information on how to specify a skinned surface.

Skinned Shell

Beta Capability
The skinned shell capability is under active development and should be used with caution.

Occasionally skinning existing elements is not sufficient, and one may require new elements/nodes to be created. The key difference between this *skinned shell* block and a *Skinned Surface* is that the *skinned shell* block creates new shell elements (and an associated shell surface) with distinct nodes that are collocated with the faces of the specified source surface, rather than creating a surface part comprised of existing element sides.

The `touching` and `excluding` syntax is the same as that of the `skinned surface`. The `intersecting` functionality is omitted, as internal faces are never included for `skinned shell` parts. Similar to above, at least one filter must be specified, and the trivial `touching blocks = all_blocks` filter can be used to skin the entire domain.

Since this creates new shell elements, one must always *specify the material* to associated with the shell elements (and underling shell surfaces). For example, the following two lines would create skinned shells with the same geometry as Fig. 4.2, with the distinction that the dashed red line would now consistent of distinct shell elements/nodes

```
# Scope: Sierra > Procedure > Aria Region
begin skinned shell skinned_all
  touching blocks = all_blocks
  use material shell_mat
end
begin skinned shell skinned_b1b2
  touching blocks = block_1 block_2
  use material shell_mat
end
```

Along with a shell block `my_shell`, the block `begin skinned shell my_shell` will also generate an associated surface with `_surface` appended (e.g. `my_shell_surface`). This surface is comprised of all exposed surfaces of the shell. See the *command reference* for more information on how to specify a skinned surface.

Interpolated Displacement

A common use case of skinned shells is to *interpolate* a displacement solution so that the shell tracks the skinned surface it was created over. This is especially useful when different components of the field are recessing at different rates, and the skin shell acts as a smooth surface tracking these interfaces to couple over. It is often helpful to specify a *Node Merge Tolerance* so that the resulting skinned surface is continuous.

Consider the example demonstrated in Fig. 4.6 with `coupling_shell` in red long dash and `contact_surface` in blue alternating dash

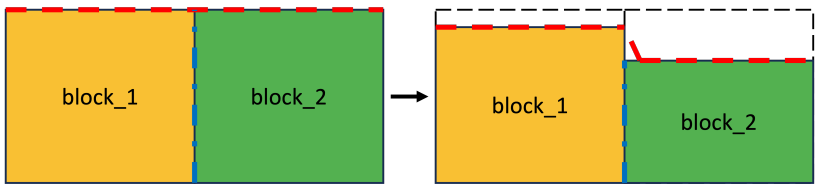


Fig. 4.6: Skinned Shell on top face of blocks with differing recession rates

In this use case, one would first define the skinned shell over the coupling surface(s)

```
# Scope: Sierra > Procedure > Aria Region
begin skinned shell coupling_shell
  touching surfaces = top_surface
  use material shell_mat

  interpolate mesh displacements from top_surface contact_surface
  node merge tolerance = 0.05
end
```

Relevant coupling quantities can then be *transferred* to/from the shell in the same *solution control* block where the shell's region is advanced e.g.

```
# Scope: Sierra > Procedure > Solution Control > System
Begin Transient MySolveBlock
  transfer shell_to_aria
  Advance myRegion
  transfer aria_to_shell
End
```

Note that these transfers must use `physical_coordinates` and not the default of `model_coordinates`, since the interpolation search must be updated as the two bodies deform

```
Begin transfer aria_to_shell
  interpolate surface nodes from myRegion to myRegion
  Send block top_surface,contact_surface to coupling_shell_surface

  Search Coordinate Field physical_coordinates State none To physical_
↪coordinates State none
  NODES OUTSIDE REGION = TRUNCATE

  Send Field solution->temperature State Old to solution->temperature␣
↪State Old
  Send Field solution->temperature State new to solution->temperature␣
↪State new
End
```

The shell can then be used as the coupling surface in e.g. an *Mpmd Coupling*.

Rebalance

Achieving an even distribution of work between MPI ranks is essential to effective parallel scalability. The decomposition methods supported in the *Finite Element Assembly block* assume that cost of computation is roughly equal for all elements in the mesh, and seek to evenly balance the number of elements between MPI ranks. In many cases this is a good assumption, and all that is necessary to achieve good parallel scalability with Aria. However in some more complicated models this assumption breaks down, and it is possible to significantly improve Aria's parallel performance by performing one or more rebalance operations that can leverage timing information during the simulation to improve the parallel decomposition. This is particularly common in models containing features like:

- Different equations solved on different mesh blocks
- Chemistry calculations present on only a subset of the mesh, or where there is a reaction front that moves with time during the run.
- Multiple equation systems
- *Mesh adaptivity* (either standard AMR or CDFEM)

In some cases > 2x speedups can be achieved via appropriate use of dynamic rebalance. There are two recommended options for enabling rebalance depending on the features in a model:

1) For cases without transient adaptivity, CDFEM, or chemistry calculations with a moving reaction front a single rebalance operation run after 3-5 time steps is often sufficient. This can be achieved via a few line commands at the Aria Region scope:

```
Begin Aria Region myRegion
  Enable Rebalance
  Rebalance Time Step Frequency = 3
  Maximum Number of Rebalances = 1

  ...
End
```

2) For more dynamic cases with transient adaptivity, CDFEM, or some chemistry calculations it can be advantageous to rebalance multiple times during the run. Note that each rebalance operation has some associated cost so rebalancing too frequently can hurt performance. For example, the input below will check if a rebalance is needed every 10 time steps, and do the rebalance if the mesh imbalance is more than 1.2. The default threshold if the `with threshold = X` argument is omitted is 1.25:

```
Begin Aria Region myRegion
  Enable Rebalance with threshold = 1.2
  Rebalance Time Step Frequency = 10
```

(continues on next page)

(continued from previous page)

```
  ...
End
```

3) When using *adaptivity*, you can also enable a rebalance to happen after every mesh adapt, using:

```
Begin Aria Region myRegion
  Rebalance after adaptivity

  ...
End
```

A full listing of the commands for controlling rebalance is provided *here*

Note: The issue with creating *multiple output files* that applies to transient adaptivity also applies to rebalance.

Adaptivity

Aria supports several forms of mesh refinement:

- Uniform refinement prior to the start of the simulation.
- Transient refinement and un-refinement based on a marker field during a simulation
- Transient conformal decomposition and refinement (TODO ref CDFEM + LS section)

Adaptivity Overview

Setting up a problem with mesh adaptivity generally requires defining three things:

1) **Indicators** - Indicators are element fields that you define to use for deciding which elements should be refined and which should be un-refined.
2) **Markers** - Once you have defined any required indicators, the next step is to define markers that use those indicators and mark elements for refinement or unrefinement. When defining multiple markers, refinement takes precedence over un-refinement, so if any marker marks an element for refinement, it will be refined even if another marker marks it for un-refinement.
3) **Solution Control** - Finally, you must use solution control to control when you want adaptivity to be active (with the exception of simple uniform refinement, which Aria offers a shorthand syntax for). Solution control allows you to adjust the timing and frequency of the

adaptivity - for example, turning it off during an initial time period, and on later in the simulation.

The following sections will show how to use these three components to set up flexible adaptivity problems. For the complete list of adaptivity commands, check the *adaptivity command reference*.

Uniform Refinement

When using only uniform initial refinement, there is a shorthand syntax available to simply refine all elements in the mesh. This can be requested in the Aria Region without setting up any indicators, markers, or solution control steps

```
Begin Aria Region myRegion
  Initial Uniform Refinement for 2 iterations

  ...
End
```

Alternately, you could use a uniform marker with a max level of 2 and use solution control to achieve the same thing. The advantage of this approach is that can be extended to define different uniform refinement markers on different blocks, each with its own maximum refinement level. For example, to use uniform refinement of 2 levels on all blocks except block_3, where you need 3 levels of uniform refinement, you could do:

```
Begin adaptivity options my_adapt
  begin uniform marker my_marker_a
    maximum refinement level = 2
    location = all_blocks
  end
  begin uniform marker my_marker_b
    maximum refinement level = 3
    location = block_3
  end
End

# in solution control
Begin System Main
  # Start by running the adapt loop some number of
  # steps (equal to the max refinement level)
  Begin AdaptiveLoop adaptive_loop
    Adapt mesh for MyAriaRegion using my_adapt
  End
```

(continues on next page)

(continued from previous page)

```
  # Then run a transient case without adaptivity
  Begin Transient TimeBlock
    Advance MyAriaRegion
  End
End

Begin Parameters For AdaptiveLoop adaptive_loop
  Number Of Adaptivity Steps Is 3
End
```

Note: Note that restart is fully supported for adaptivity. For initial refinement, though, one should modify the given input to ensure that the adaptive loop is not executed on restart runs

```
Begin AdaptiveLoop adaptive_loop
  Adapt mesh for MyAriaRegion using my_adapt when "current_
↪time == 0.0"
End
```

Runtime Adaptive Refinement

Defining Markers and Indicators

To use runtime adaptivity, you will typically want to define a more complex set of indicators and markers. These could include uniform refinement on some blocks, and one or more indicators or markers on other blocks.

```
Begin Adaptivity Options my_adapt
    Begin signed distance indicator ls_distance
        signed distance field = solution->Level_Set
        location = block_1
    End

    # Mark elements on block_1 less than 0.05 from the
    # level set interface to refine 2 levels
    Begin function marker ls_marker
        refine when "ls_distance < 0.05"
        unrefine when "ls_distance >= 0.05"
        Maximum refinement level = 2
        location = block_1
```

(continues on next page)

(continued from previous page)

```
    End

    # Mark all elements inside a geometric shape to be refined
    # 3 levels across two blocks
    Begin function marker plug_marker
        refine when "(z > 0) && (sqrt(x^2 + y^2) < 0.3)"
        Maximum refinement level = 3
        location = block_2 block_3
    End

    # Mark all elements on block_2 to be refined 1 level
    begin uniform marker case_marker
        maximum refinement level = 1
        location = block_2
    end
End
```

Solution Control Integration

To enable a single adapt before each time step, add an `Adapt Mesh` command inside the `Begin Transient` block.

```
Begin System Main
  Begin Transient TimeBlockMain
    Adapt Mesh for MyAriaRegion Using my_adapt
    Advance MyAriaRegion
  End
End
```

The use of an `AdaptiveLoop` block allows multiple passes of mesh refinement per time step. To enable multiple adaptivity passes before each time step, add a nested `AdaptiveLoop` block to run multiple back-to-back adaptivity cycles at the start of each time step.

```
Begin System Main
  Begin Transient TimeBlockMain
    Begin AdaptiveLoop adaptive_loop
      Adapt mesh for MyAriaRegion using my_adapt
    End
    Advance MyAriaRegion
  End
End
```

(continues on next page)

(continued from previous page)

```
Begin Parameters For AdaptiveLoop adaptive_loop
  Number Of Adaptivity Steps Is 2
End
```

Alternately, you may want each time step to take multiple Adapt-Step iterations. To achieve that, move the `Advance` inside the adaptive loop. When using this combination, it will repeat the adapt-step iteration at that same time step how ever many times you have requested. This may provide better error control since the adaptivity indicators can use the updated solution fields, but comes at the cost of evaluating additional time steps with their full linear assembly and solve. Additionally, you can include an extra `Advance` call before the adaptive loop so that the first adapt step uses the updated solution fields.

For example, if the current time step is from 2 s to 3 s, the adaptive loop below (with 2 iterations) would:

1. (optional initial Advance) Step from 2 to 3 s and update the solution fields
2. Calculate indicators and markers using the current (updated or from the prior step) solution fields and adapt the mesh
3. Step from 2 to 3 s and update the solution fields
4. Calculate indicators and markers using the updated solution fields and adapt the mesh
5. Step from 2 to 3 s and update the solution fields

If there were more than 2 iterations requested, it would continue to repeat this cycle as many times as requested. The solution control input for this scheme would look like:

```
Begin System Main
  Begin Transient TimeBlockMain
    # Optional: initial advance so first adapt uses this time step␣
↪solution
    Advance MyAriaRegion

    Begin AdaptiveLoop adaptive_loop
      Adapt mesh for MyAriaRegion using my_adapt
      Advance MyAriaRegion
    End
  End
End

Begin Parameters For AdaptiveLoop adaptive_loop
  Number Of Adaptivity Steps Is 2
End
```

Note: When doing adaptive loops within a time step, advancing the region multiple times in a single time step may not be desirable due to computational expense. If the indicator field is a function only of postprocessed values (such as a source in an equation system), the adaptive loop can include only a postprocessor call e.g.

```
Begin System Main
  Begin Transient TimeBlockMain
    Begin AdaptiveLoop adaptive_loop
      Postprocess Aria region MyAriaRegion
      Adapt mesh for MyAriaRegion using my_adapt
    End
    Advance MyAriaRegion
  End
End
...
Begin Aria region MyAriaRegion
  ...
  EQ Energy for Temperature on all_blocks using Q1 with lumped_
 ↪mass Diff src
  Source for Energy on all_blocks = Scalar_String_Function F =
 ↪"..."

  postprocess value of expression energy_source on all_blocks␣
 ↪as source_field
End
```

Adaptivity Considerations

There are a few usability inconveniences to be aware of when enabling transient adaptivity:

- When using adaptivity, it is usually necessary to activate *rebalance* to achieve good performance, otherwise some ranks can end up with substantially more work than others.
- Due to limitations in the Exodus file format each time the mesh changes Aria will create a new output file. These output files will have "-sNNNN" appended to the initial file name where "NNNN" is an increasing number corresponding to the order of the files. So for example if the output database name was initially "output.e" then a run where adaptivity was performed 3 times would have 4 output files: "output.e", "output.e-s0001", "output.e-s0002", and "output.e-s0003". If running in parallel and the default file-per-processor output format is being used a new one of these files will be created for every MPI rank. Fortunately visualization tools like Paraview understand this file name convention and will visualize the full set of files as a single result, but the number of files created in long-running simulations with adaptivity can make file management a nuisance.

- Each time the mesh is modified during the simulation run Aria needs to reinitialize a number of data structures that are associated with the mesh. Most notable of these is that (depending on viewfactor settings and the specific elements refined) the viewfactors may need to be recomputed. This can significantly increase the cost of simulations using adaptivity with enclosure radiation. See *here* for strategies on managing this.

4.3 Problem Specification

Equation Specification

This chapter documents the EQ line commands within the current version of Aria. EQ line commands add equations and independent variables to Aria's specification of the equation set to be solved for within each region. The equation is also associated with a field variable here that becomes part of the solution vector for Aria. EQ commands occur in Aria's input file within Region blocks.

Note: For more information on the equations aria solves, refer to the *theory* section. A comprehensive list of `EQ` line commands can be found in the *command reference*.

The EQ command line adds an equation to be solved for on a particular `MESHPART`. The format is as follows

```
EQ [equation] FOR [DOF] on [MESHPART] using [INTERP] with TERM$_0$ ...␣
↪TERM$_n$
```

`Equation` is the string identifier for the individual equations listed in *General Naming Convention*. Namely the material phase, species, or LS phase can be specified in the `Equation` argument e.g.

```
# Equation for solid phase voltage
EQ current for voltage in solid_phase on block_1 ...

# Equation for Li+ in liquid phase
EQ species for species of Li+ in liquid_phase on block_1 = ...
```

The `DOF` keyword specifies the independent unknown that is solved for in order to satisfy the equation. Normally, it is a strict function of the equation keyword. In other words, the `temperature` is the only valid `DOF` entry if the energy equation is being solved. `MESHPART` is usually the name of an active element block in the finite element model. Unfortunately, if an equation is to be solved on the entire finite element model, this means that there must be multiple EQ keywords for each element block defined in the mesh. Alternative an assembly or meshgroup can be used (see *Assemblies and Mesh Groups*).

INTERP defines the finite element interpolation to be used. Currently the valid entries for this keyword are P0, P1, Q1, Q2 and Q2S with standard volumes, e.g. hexahedron, tetrahedron, wedge, pyramid, quadrilateral, triangle and bar. These keywords imply the interpolation and polynomial order for the solution field where P denotes (polynomial) element interpolation and Q denotes the nodal interpolation - Q1: linear - Q2: quadratic - Q2S: near-quadratic , serendipity

We note that no syntactical distinction is made between element topology (i.e. triangle and quadrilateral or tetrahedron and hexahedron), that is, the syntax indicates only the level of interpolation. The association of element topology with the equation is handled internal to the code and allowing the specification of interpolation independent of the topology enables the use of superparametric elements (i.e. quadratic geometry with linear interpolation).

Parametric coordinates of shell and bar elements differ from those of conventional bulk volume elements (e.g. hexahedron and tetrahedron) since their meshed geometric description lacks a thickness or area dimension. This missing dimension (SHELL THICKNESS and BAR AREA) can be provided for each material independent of the meshed discretization. Hence when defining equations on shell elements the INTERP keywords previously mentioned are paired (e.g. Q1P0) according to interpolation in the shell plane and interpolation through the shell thickness. For example Q1P0 represents linear nodal interpolation in the plane of the shell and constant interpolation through the shell thickness. Similarly for bar elements an INTERP specification of Q1P0 represents linear nodal interpolation along the length of the bar and constant interpolation in the bar section. In sets of coupled equations, arbitrary combinations of interpolations are sometimes not permitted for solution variables appearing in coupling terms of an equation as interpolation must be consistent in any given equation.

TERM_n refer to the broad categories for the terms in a general advection-diffusion continuity equation. Each term in the equation must be explicitly turned on for it to appear in the conservation equation. Admissible values of TERM are MASS, LUMPED_MASS, ADV, DIFF, SRC, and XFER. While MASS, LUMPED_MASS, ADV, DIFF, SRC terms imply the activation of mathematical operators appearing in an equation, XFER implies that a solution field values for this equation are being provided externally so the equation can still be thought of as belonging to the overall equation set.

Provisions are made for supplying multiple equation contributions for DIFF. These contributions are usually activated via settings in the Aria Materials command block (see *Material Properties*). Multiple contributions can also be defined for SRC by adding additional source command lines (see *Volumetric Sources*). In order to accommodate different definitions of velocity for ADV the user can request the velocity to be provided from another equation, from transfer or from other *available velocity models*. See also *Advection Velocity*.

All equations are assumed to be formulated in Cartesian coordinates. For two-dimensional problems, axisymmetric formulations are available as well as the Cartesian forms. Invocation of the axisymmetric option is described in the Model Definition chapter in *Finite element model*.

Material Properties

In order to solve the equations specified above, properties of the material(s) which make up the domain must be defined. In Aria, this is accomplished by creating material definitions which are assigned to their corresponding portions of the domain (blocks, surfaces, etc). A material is assigned to a specific mesh element using the following

```
# Scope: Sierra
Begin Finite Element Model The_Mesh
  # Other finite element model commands

  use material Kryptonite for block_1
  use material Kryptonite_surf for surface_1
End Finite Element Model The_Mesh
```

See *Finite element model* for more information on specification of the finite element model.

Material property values are prescribed for unique material names using input line commands within an `Aria Material` command block. The general format of a material property line command is as follows:

[PROPERTY] = [MODEL] [MODEL ARGS...]

This line populates the `PROPERTY` expression using `MODEL` on blocks for which the given material is assigned (For more details on the expression system employed by Aria, refer to *Expression Overview*). As a demonstration, consider the following geometry:

```
  surface_1
+-----------+
|  block_1  |
+-----------+
```

with materials:

```
# Scope: Sierra
Begin Aria Material Kryptonite
  #! 1)
  Emissivity = constant e=0.7

  Begin Material Phase gas_phase
    Species Names = speciesA speciesB speciesC

    #! 2)
    Density = general_ideal_gas R={Rgas}

    #! 3)
```

(continues on next page)

(continued from previous page)

```
    molecular_weight of speciesA = constant M={MspeciesA}
    molecular_weight of speciesB = constant M={MspeciesB}
    molecular_weight of speciesC = constant M={MspeciesC}

  End Material Phase gas_phase
End Aria Material Kryptonite

Begin Aria Material Kryptonite_surf
  #! 4)
  heat transfer coefficient = SCALAR_STRING_FUNCTION \$
     f = "2*Emissivity"
End Aria Material Kryptonite_surf
```

This example illustrates the following cases:

1. In the most general declaration, the property `Emissivity` is defined using the `Constant` model with argument `E=0.7`. This will produce the expression `EMISSIVITY` on all blocks which have material `KRYPTONITE` assigned.
2. In multi-phase problems, properties can depend on the phase of material. This line definition declares the `DENSITY` of the `GAS_PHASE` of `KRYPTONITE`, and will produce the expression `GAS_PHASE_DENSITY`.
3. Finally if multiple species are present, the property of individual species can be specified by appending the property with the respective species (i.e. `PROPERTY = MOLECULAR_WEIGHT OF SPECIESC`). This will produce an expression `GAS_PHASE_MOLECULAR_WEIGHT_SPECIESC`.
4. Note that children of a part inherent the material properties of their parent (e.g. surfaces inherent properties from their owning block). This means that since `SURFACE_1` is connected to `BLOCK_1`, it will have e.g. `EMISSIVITY` defined via `BLOCK_1`'s material definition. The opposite is *NOT* true, meaning `BLOCK_1` will *NOT* have `HEAT_TRANSFER_COEFFICIENT` defined.

The reader is referred to the *command reference* for a comprehensive list of all possible material property definitions and model options.

Note: Chemical material models solved using the `CHEMEQ` library require an entire set of parameters. In this case those parameters are defined within a `Parameters for CHEMEQ` command block embedded within the `Aria Material` command block. A description of the CHEMEQ related parameters can be found in *CHEMEQ Reference*.

Note: Some tensor properties which are often isotropic can be defined as a single scalar for efficiency. Individual tensor components can be specified by prepending `TENSOR_` to the property name in the material line definition. Otherwise, the scalar case corresponds to $\alpha \boldsymbol{I}$. See *Thermal Conductivity* vs *Tensor Thermal Conductivity* for an example of this.

Volumetric Sources

The Source line applies a model for a volumetric source term for a given equation. See *Governing Equations* for more information on the role of sources in Aria. The general format of a source line command is as follows:

`Source for` *`[EQUATION]`* `on` *`[PART] = [MODEL] [MODEL ARGS...]`*

Here the `EQUATION` must be one declared with an `EQ` line including a `SRC` term defined at the same scope (i.e. Equation System or Region scope, see *Equation Specification*).

The `PART` corresponds to a mesh entity, and can include both single parts (e.g. *block_1*) as well as assemblies or mesh groups (see *Assemblies and Mesh Groups*).

Finally the `MODEL` defines how the source term is calculated. Note that multiple source lines may be defined for a given `EQUATION` / `PART`.

For example, a constant source for the energy equation can be defined as follows

```
# Scope: Sierra > Procedure > Aria Region
EQ Energy for Temperature on block_2 using Q1 with DIFF SRC
Source for energy on block_2 = constant value = 0.2
```

The reader is referred to the *command reference* for a comprehensive list of all possible equations and model options.

Initial Conditions

This section documents the initial condition, IC, line commands within the current version of Aria. The IC simply represents an initialization of the Field variable. Technically speaking initial conditions have meaning only in time-dependent partial differential equations. However, within the Aria code the IC command line results in an initialization of Field values used in the assembly of a Jacobian matrix. Thus the IC is equally applicable for assigning starting values in a steady solution.

Within the input file initial conditions command lines will appear at the Region scope as illustrated below. When problem physics within the Region are defined using Equation Systems, initial conditions must be defined outside the scope of the Equation Systems. Note that input

specification of initial conditions on portions of the FEM mesh can be simplified by aggregation of element blocks into *assemblies and mesh groups*.

The general format of an initial condition line command is as follows:

`IC for [DOF] on [PART] = [MODEL] [MODEL ARGS...]`

Here the `DOF` must be one associated with an equation declared with an `EQ` line at either `Aria Region` or `Equation System` scope (see *Equation Specification*). Note that material phase, species, or LS phase can be specified in the `DOF` argument e.g.

```
# IC for gas phase temperature
IC for temperature in gas_phase on block_1 = ...

# IC for Li+ in liquid phase
IC for species of Li+ in liquid_phase on block_1 = ...
```

See *General Naming Convention* for more information.

The `PART` corresponds to a mesh entity, and can include both single parts (e.g. *block_1*) as well as assemblies or mesh groups (see *Assemblies and Mesh Groups*).

Finally the `MODEL` defines how the source term is calculated. Note that multiple source lines may be defined for a given `DOF` / `PART`.

For example, a constant IC for temperature can be defined as follows

```
# Scope: Sierra > Procedure > Aria Region
IC for temperature on block_1 = constant value = 300 # K
```

The reader is referred to the *command reference* for a comprehensive list of all possible initial condition commands.

Boundary Conditions

Boundary conditions are needed in order to define a well-posed boundary value problem as described by the FEM equations. These boundary conditions are often denoted as Dirichlet, Neumann, Robin or distinguishing conditions. Dirichlet or essential boundary conditions for a solution field stem directly from the governing equations for a particular physics. In Aria, Neumann and Robin conditions are generally categorized as flux boundary conditions. Distinguishing conditions enforce a boundary i condition via an auxiliary equation, which is discussed in more detail in *Disting*.

This section documents primarily the native Aria boundary condition, BC, line commands within the current version of the code. Recall from a previous chapter outlining the general FEM that the method gives rise to boundary terms *Governing Equations*, i.e. various flux boundary conditions. In the command reference, Dirichlet boundary conditions are described as well as the various flux boundary condition types.

Boundary conditions are defined at the Region scope of the input file. See the *command reference* for a comprehensive list of possible BC options. The general format of a boundary condition line command is as follows:

`BC [TYPE] for [QUANTITY] on [PART] = [MODEL] [MODEL ARGS...]`

Here the `TYPE` denotes if the BC is a dirichlet BC, one of the flux BC types, or disting (distinguishing).

The `QUANTITY` is defined as follows:

- For `BC Dirichlet`, the quantity is the `DOF` field on which the BC is applied. This `DOF`
- Otherwise, the quantity is the `EQUATION` contributed to by the flux or distinguishing condition.

In either case, the equation/DOF must be declared with an `EQ` line at either `Aria Region` or `Equation System` scope (see *Equation Specification*). Note that material phase, species, or LS phase can be specified in the `DOF/EQUATION` argument e.g.

```
# Dirichlet BC for gas phase energy equation
BC flux for energy in gas_phase on block_1 = ...

# Dirichlet BC for Li+ in liquid phase
BC dirichlet for species of Li+ in liquid_phase on block_1 = ...

# Distinguishing BC for no slip mesh motion, with zero "leak" velocity
BC Disting for Mesh_x on surface_2 = Kinematic v0=0
```

See *General Naming Convention* for more information.

Boundary conditions can also be specified as string functions (see *String Functions*), which provides a mechanism for user customization. For example, a user can define a distinguishing condition that is only active in regions above a certain temperature, t_ref, and otherwise does not alter the solution as

```
BC Disting for Species on surface_2 = Scalar_String_Function
  F = "(temperature - t_ref) > 0.0 ? (species - 100) : 0.0"
```

The `PART` corresponds to a mesh entity, and can include both single parts (e.g. *block_1*) as well as assemblies or mesh groups (see *Assemblies and Mesh Groups*).

Finally the `MODEL` defines how the source term is calculated. Note that multiple source lines may be defined for a given `DOF` / `PART`.

For example, a dirichlet BC on temperature and flux BC for the energy equation can be defined as follows

```
# Scope: Sierra > Procedure > Aria Region
BC dirichlet for temperature on surface_1 = constant value = 300 # K
BC flux for energy on surface_2 = constant flux = 1.0
```

Periodic BCs

Aria provides a periodic boundary condition capability. Two flavors of periodicity are supported, standard periodicity for surfaces separated by fixed distance and cyclic periodicity for surfaces separated through a fixed rotation angle as illustrated in Fig. 4.7. The underlying implementation is based upon augmenting the basic governing equations with multi-point constraints on nodal DOF at the periodic boundaries and is equally applicable to scalar and vector DOF. The enforcement employs nodal DOF, hence the capability is limited to grids with conforming surface nodes, i.e. the mesh is periodic as well. Note that for non-conforming surface topologies, *periodic contact* can be used instead.

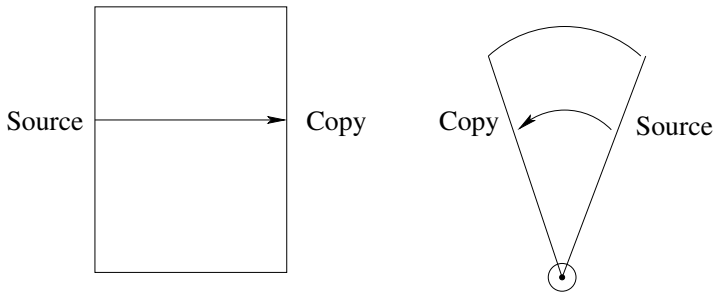


Fig. 4.7: Standard and Cyclic Periodicity

Internally, a search procedure is used in order to define node constraint pairs. Since a meshed discretization is subject to intrinsic error in the nodal coordinates, an acceptable tolerance specification is used to insure that pairings are appropriately defined. Note that application of large tolerances may lead to incorrectly defined pairings.

See the *command reference* for more details on specifying periodic BCs. Note that cyclic periodicity requires that the user specify both *Reference Axis* of rotation and *Point on Axis* (a point on the reference axis) in the *Periodic* command block. Here `Reference Axis` and `Point on Axis` are associated with *Define Direction* and *Define Point* respectively. These two additional quantities are usually defined at the Domain scope (outside of the Procedure).

Coupling BCs for Organic Material Decomposition

Aria provides the capability of coupling between porous (see *Organic Material Decomposition (OMD)*) and fluid blocks using flux-style boundary conditions. The boundary conditions described here are for a single region computation; for the most part, these boundary conditions can be used for a multi-region computation with the phrase 'one_region' omitted. These boundary conditions are imposed as Robin boundary conditions,

$$F_P = \rho_F \mathbf{u}_F \cdot \mathbf{n} + \beta(p_g - p_F)$$
$$F_F = \rho_g \mathbf{u}_g \cdot \mathbf{n} + \beta(p_F - p_g)$$

where F_P is the flux into the porous region and F_F is the flux into the fluid region. The boundary conditions for the pressure mass balance equation on the porous side and the corresponding flux condition on the fluid side CVFEM continuity equation are written as (see *Fluid_Robin_Coupled_One_Region* and *Porous_Robin_One_Region* for syntax specific details)

```
BC flux for mass_balance in gas_phase on surface = fluid_robin_coupled_
↪one_region
BC flux for cvfem_continuity on surface = porous_robin_one_region
```

This boundary condition uses a penalty coefficient of the form

$$\beta = \frac{c_{scaling} \rho_g K}{\mu_g h}$$

where h is the mesh width adjacent to the interface (need to check what happens with a non-uniform mesh). The default value of $c_{scaling}$ is 0.001. If this parameter is too large, the increase in pressure might be underpredicted.

A distinguishing boundary condition is used on the fluid momentum equation,

$$u_j - (u^D_{j,n} + u^D_{j,t}) = 0, \tag{4.1}$$

where $u^D_{j,n}$ is the imposed normal component of velocity and $u^D_{j,t}$ is the imposed tangential component of velocity. The normal component is computed directly from the continuity flux at the interface,

$$u^D_{j,n} = \frac{F_F}{\rho_F} n_j. \tag{4.2}$$

The tangential component is based on a variation of the classical Beavers-Joseph-Saffman condition [36, 37] for the slip velocity which has been extended to non-planar surfaces in multidimensional flow [38], which defines a provisional model velocity

$$u^{\mathrm{BJS}}_j = -\frac{\sqrt{\bar{K}}}{\alpha \mu} \left(n_i \tau_{ij} \right) \tag{4.3}$$

where $\bar{K}$ is the permeability of the porous region at the interface, μ is the viscosity of the local fluid at the interface, τ_{ij} is the viscous stress tensor of the fluid at the interface, and α is a

dimensionless model parameter that is a function of the microstructure of the porous material, which has been found to have typical values near 0.1 [37]. The tangential component of this vector quantity is used as the tangential component of the distinguishing condition velocity, and is computed as

$$u^D_{j,t} = u^{\text{BJS}}_j - \left(u^{\text{BJS}}_k n_k\right) n_j. \tag{4.4}$$

which is written in the input file as

```
BC Disting for cvfem_momentum on surface = porous_robin_one_region
```

The flux of a species k across a porous-fluid interface is

$$\mathbf{J}^{Y_k}_P = \mathbf{J}^{diff}_F + F_P \rho_g Y_{k,PF} + \left(\frac{D_k \bar{\rho}_f}{h}\right)(Y_{k,P} - Y_{k,F})$$

where P refers to porous side, F refers to the fluid side and $Y_{k,PF}$ is the upwinded interface mass fraction, equivalent to h_{PF} from the enthalpy coupling. The mass fraction coupling boundary condition on the porous side is written as

```
BC flux for mass_balance in phase on surface = fluid_robin_coupled_one_
↪region
```

The enthalpy coupling boundary conditions in the porous region are (see *Fluid_Robin_Coupled_With_Solid_Convection_One_Region* and *Fluid_Solid_Convection_Coupled_One_Region* for syntax specific details)

```
BC flux for porous_enthalpy in phase on surface = fluid_robin_coupled_
↪with_solid_convection_one_region
BC flux for porous_enthalpy in phase on surface = fluid_solid_convection_
↪coupled_one_region
```

4.4 Advanced Features

This section contains details on more advanced features that can be used in Aria.

Radiation

This section discusses defining radiation boundary conditions whose formulations depend upon the Stefan Boltzmann constant. Provision is made for all radiation BC models to access a single value of the constant from *command reference*.

Enclosure Radiation

As described *here*, enclosure radiation accounts for radiative heat transfer between surfaces of an enclosure wherein the intermediate medium is transparent. Modeling of enclosure radiation requires determination of the energy transport between all N surface facets that comprise the enclosure, an N^2 description of global interactions. Since direct coupling of the N^2 radiosity problem with the volume energy equations would result in a monolithic solve, the approached used in Aria is instead to solve the enclosure radiation and volume energy equations (as well as any other equations that may appear in the equation system) independently but coupled through boundary conditions. Specifically, the enclosure radiation problem results in the following three main steps:

- Compute the viewfactors (if needed) or read in pre-computed viewfactors (if requested)
- Solve the radiosity equation (3.114) using the current average facet temperature to compute a known RHS
- Solve the linearized equation system for the temperature correction (as well as other DOF corrections) using the updated radiative heat flux (3.115) contribution and correct the temperature.

where the viewfactor step occurs prior to the nonlinear loop, whereas the last two steps repeat until the specified nonlinear convergence criteria is met.

Note: A given enclosure can be coupled through its surfaces to multiple energy equation types (e.g. `EQ ENERGY`, `EQ POROUS_ENTHALPY`), that may belong to different equation systems. When this occurs, the viewfactor computation and radiosity solve are only performed in the first (input deck order) segregated equation system that is coupled to the enclosure. As a result, the radiosity solve specified by (3.114) will use lagged facet temperatures from other equation systems which will result in lagged radiosities and correspondingly a lagged radiative flux will be applied. To avoid this, it is recommended that respective equations are all defined in a single equation system.

The input deck below shows the incremental change needed to add an enclosure.

```
Begin Sierra myJob

  ...

  Begin Global Constants
    Stefan Boltzmann Constant = 5.6704e-08  # W/m2-K4
  End

  Begin Aria Material my_material
```

(continues on next page)

(continued from previous page)

```
  # Define emissivity model
  emissivity = constant value = 1.0
  ...
End

Begin Procedure My_Aria_Procedure
  ...
  Begin Aria Region My_Region
    ...

    # Define the enclosure
    Begin Enclosure Definition enc1
      add surface surface_1 surface_2

      # Uncommenting will override the above emissivity model
      # emissivity = 0.5 on surface_1
      # emissivity = 0.6 on surface_2

      Use Viewfactor Calculation vf_calc
      Use Viewfactor Smoothing no_smooth
      Use Radiosity Solver Rad_Solv
    End

    # Define how the viewfactor is calculated
    Begin Viewfactor Calculation vf_calc
      Compute Rule            = Hemicube
      ...
    End

    # Define viewfactor smoothing
    Begin Viewfactor Smoothing no_smooth
      Method                = none
      ...
    End

    # Define solver to use for the radiosity equation
    Begin Radiosity Solver Rad_Solv
      Solver                  = chaparral GMRES
      Convergence Tolerance   = 1e-8
      Maximum Iterations      = 300
      ...
    End
```

(continues on next page)

(continued from previous page)

```
      ...

      postprocess average of expression irradiance on surface_1 as G_avg
      postprocess average of expression rad_flux on surface_1 as q_avg
      postprocess average of expression radiosity on surface_1 as J_avg

      # Define enclosure facet field variables for output
      Begin Results Output Label encl_rad
        ...
        # in 2D, these are edge variables
        face variables = radiosity as J
        face variables = rad_flux as q
        face variables = irradiance as G
        face variables = emissivity as e
        face variables = face_temperature as T
      End
    End
  End
End Sierra myJob
```

To start, an enclosure definition command block defines the surfaces of the enclosure. These surfaces are automatically coupled to respective volume energy equations through a radiative heat flux contribution. On these surfaces, several enclosure specific face field variables are available for output, these include the `rad_flux` (3.115), `irradiance` (3.111), `radiosity` (3.109), `emissivity`, and `face_temperature` (3.110). As shown above, expressions for the `rad_flux`, `irradiance`, and `radiosity` are also provided so that reductions operations can be performed.

Note: A dirichlet boundary condition for temperature specified on one of the enclosure surfaces will override any flux contribution from the enclosure to that surface. The radiosity equation will still be coupled to this surface through its right hand side term.

Specification of enclosure surfaces can either be done directly by enumerating the surfaces as shown above or alternatively, when used with the *dash enclosures capability*, by specifying a combination of volume blocks to skin and surfaces to include.

Warning: By default, if an enclosure surface touches a volume block that only has `EQ POROUS_ENTHALPY` in either the solid phase or gas phase, the radiative flux will only contribute to the solid phase. When it is ambiguous which energy equation to couple the enclosure to, an error will be thrown. To resolve the error, the *matched flux line command* must be used to specify which equation should receive the radiative heat flux.

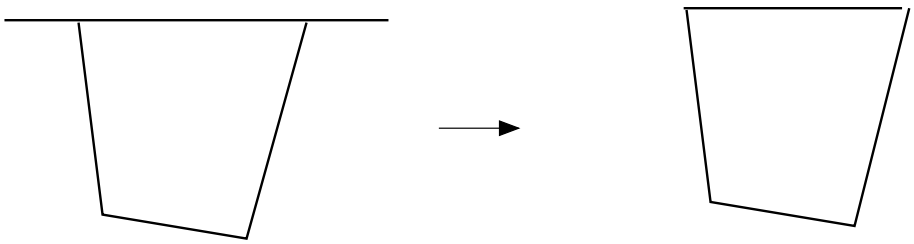


Fig. 4.8: Discretization and Associated Closed Enclosure Surface.

In all cases, the enclosure is viewed as a closed surface. As illustrated in Fig. 4.8 the discretization may need to be modified in order to properly define the closed surface. For problems in which the enclosure surface is not entirely meshed, it can be implicitly closed by defining a *partial enclosure* with appropriately defined properties specified within the enclosure definition command block.

As shown in the example input deck, within each enclosure definition command block, line commands reference separate command blocks that provide details concerning the viewfactor calculation, viewfactor smoothing, and radiosity solver.

Note: Line commands can be also used to define the emissivity for a particular surface in the enclosure definition command block. When the emissivity has been provided both from the block material (as shown above) and surface material, the surface material will override.

With respect to the viewfactor calculation, we have specified the most common approach i.e., the hemicube method. Details regarding the hemicube method are provided in [19]. The *viewfactor command reference* details additional line commands relevant to the hemicube method. If the same geometric model will be used for different analyses, saving the viewfactors to file and reading them in is recommended to reduce simulation time. This can be accomplished by first saving the viewfactors to a file via a command in the enclosure definition command block and then in a subsequent run changing the compute rule to read in the specified viewfactors e.g.

```
Begin Enclosure Definition enc1
  ...
  database name is viewfactors.vf in pnetcdf format
End

Begin Viewfactor Calculation vf_calc
  compute rule = read
  ...
End
```

Note: If no viewfactor file is found, the compute rule will default to the hemicube method. A message will be printed to the log indicating that viewfactors are being computed as opposed to read.

With respect to viewfactor smoothing, the above example uses none; however, complex enclosures often require smoothing of the viewfactors to ensure the quality of the viewfactors before proceeding to the radiosity solve. One measure of how accurately the viewfactors are being computed is the rowsum value. For each surface facet i in a fully-closed enclosure the rowsum value should approach one i.e., $\sum_{j=1}^{N} F_{ij} = 1.0$, thus the total rowsum value is always the number of facets. In cases where the target rowsum is not equal to one the surface may not be fully-closed or the viewfactor calculation may be inaccurate for some other reason. In either event the analyst should investigate possible causes of the discrepancy. Viewfactor smoothing may offset some of the errors indicated by the rowsum values. The viewfactor smoothing command reference can be found *here*.

Note: The viewfactor between two facets is roughly proportional to $1/r^2$ where r is the distance between facet centroids. Thus it follows directly that facets which are nearly parallel and opposing may adversely affect the viewfactor calculation. Similar statements apply to poorly equivalenced surface intersections leading to slivered surfaces becoming part of an enclosure. In the worst of circumstances the viewfactor will simply fail when the distance between facets is too small and the mesh may need to be repaired.

The above example outlines input commands blocks for enclosure radiation problems that are independent of the wavelength. When considering the wavelength dependence, a banded wavelength model can be used. This requires specifying additional command blocks as discussed *here*.

Note: For most cases the enclosure is devoid of interior mesh discretization. In the event that the enclosure interior is meshed, the interior mesh can be ignored by using the `OMIT_VOLUME` command line in the FINITE_ELEMENT_MODEL command block. In the event one might require the interior mesh to represent other coupled physics within the cavity area of the enclosure, the `MESHED_ENCLOSURE` command can be specified in the enclosure definition command block to exclude the meshed block (and its surface) from the enclosure radiation problem. For simulations in which enclosure radiation approaches the optically-thick limit one might consider utilizing a meshed enclosure in conjunction with the `OPTICALLY\_THICK` thermal conductivity model in lieu of the enclosure radiation problem.

A full command reference for the various relevant line commands for all enclosure radiation command blocks is available *here*.

Partial Enclosures

When an enclosure is not entirely meshed, partial enclosures can be used to "close" the enclosure through a far field radiating condition. An example partial enclosure is demonstrated in Fig. 4.9.

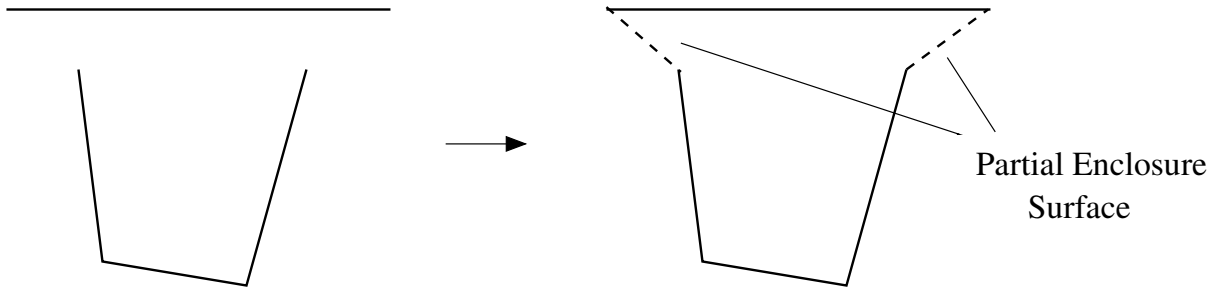


Fig. 4.9: Discretization and Associated Partial Enclosure Surface

Warning: It is important to ensure that the partial enclosure is properly formed and that unintended gaps are not included in the virtual partial surface. Notably, partial enclosures do not have row sum errors. As such, forgetting to include a surface or having a mesh defect will result in unintended energy leakage. Lastly, viewfactor smoothing can not be applied with partial enclosures as it will lead to erroneous viewfactors.

The concept of a partial enclosure is best demonstrated by use of a simplified view of radiative transfer system between surfaces 1 and 2 surrounded by free space as shown in Fig. 4.10.

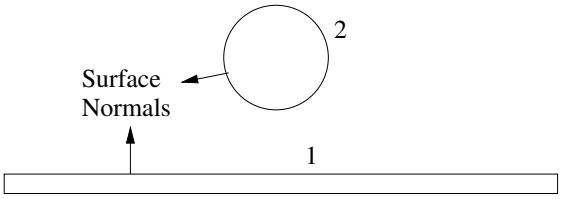


Fig. 4.10: Two-Body Radiative Transfer Model.

If radiative transfer occurs between the two bodies and the viewfactor between the two bodies is readily accessible a simplified network representation of the interaction, Fig. 4.11 may suffice but here the interest lies in analyzing the system using enclosure radiation.

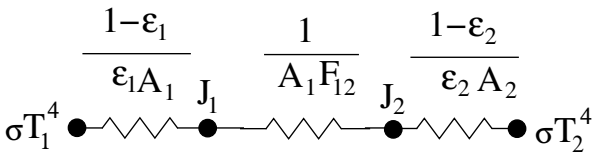


Fig. 4.11: Two Surface Network Model.

In many cases a more complete model of the two-body configuration will include radiative transfer with the surroundings. Including these interactions requires introduction of an additional enclosure surface as shown in Fig. 4.12.

Surface 3 known as the "partial enclosure" is artificially added to the numerical model (i.e. not included in the meshed discretization). Introduction of the partial enclosure is necessary in order to apply conventional approaches for numerical evaluation of the system viewfactors. Here we note that algorithmically, failure to add the partial enclosure surface would be manifest in bad row-sum metrics.

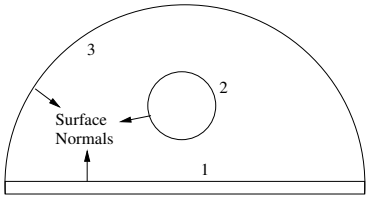


Fig. 4.12: Partial Enclosure Radiative Transfer Model.

Introduction of the partial enclosure leads to the thermal network representation shown in Fig. 4.13 in which the partial enclosure interacts with the other bodies of the system.

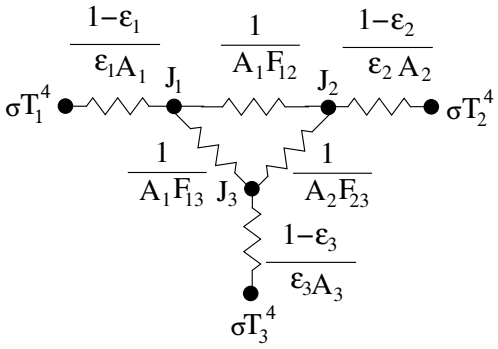


Fig. 4.13: Three Surface Network Model.

Since the partial enclosure is modeled in the same way as a real surface, questions often arise concerning characterization of the partial enclosure in terms of its temperature, area and emissivity. The temperature must of course must represent the surroundings. The area should be an area that envelopes the true surfaces of the enclosures. The partial enclosure emissivity should be chosen in a manner consistent with the relationship of surface radiosity J with the emission, σT_3^4 and irradiance, G i.e.,

$$J = \varepsilon\sigma T^4 - (1-\varepsilon)G \ . \tag{4.5}$$

As examples of this we consider the consequence of selecting a partial emissivity at two extremes, $\varepsilon = 0$ and $\varepsilon = 1$, both of which reduce the network model of Fig. 4.13 to that of Fig. 4.14.

For $\varepsilon = 0$ from the network interaction between $\sigma T_3^4 - J_3$ we find that the resistance becomes infinite thus eliminating any system interaction with T_3 as all the incident energy is reflected. This

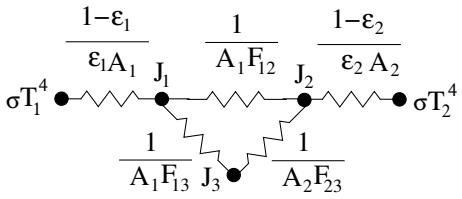


Fig. 4.14: Three Surface Network Model ($\varepsilon_3 = 0$ and $\varepsilon_3 = 1$).

results in a modified version of the two-surface network Fig. 4.11, albeit with more attenuation. For $\varepsilon = 1$ and the network interaction between $\sigma T_3^4 - J_3$ is eliminated. However, now the radiative transfer system sees the emissive power of the surroundings since from (4.5), $J = \sigma T^4$. Clearly there are no obvious choices for partial enclosure emissivity other than that it should be chosen to enable interaction of the system with the surroundings.

As part of the partial enclosure description one must supply a partial enclosure area. For many problems it may be difficult to initially compute or even estimate the partial enclosure area. For these cases provision is made within Aria to internally compute the minimum area that should be used. To obtain the minimum area one temporarily assigns a small number for the partial enclosure, starts the problem and terminates it after one step. The minimum partial enclosure area will be printed to the Chaparral log.

Note: The partial enclosure modifies the radiosity equation by one row and column with the partial enclosure properties used to compute matrix entries and the right hand side entry. For instance, the partial enclosure area determines the last row of viewfactors F_{Nj} via the viewfactor reciprocity relation i.e., $A_N F_{Nj} = A_j F_{jN}$, where the last column of viewfactors F_{iN} are derived from the row sum relation i.e., $\sum_{j=1}^{N} F_{ij} = 1.0$.

Once the partial enclosure area, temperature and emissivity have been determined, they can be specified in the enclosure definition command block as follows

```
Begin Enclosure Definition enc1
  ...
  # Define partial enclosure
  partial enclosure area = 1.5738
  partial enclosure temperature = 293.0
  partial enclosure emissivity = 1.0

  # Define partial enclosure variables for output
  partial enclosure flux output partial_flux
  partial enclosure radiosity output partial_radiosity
  partial enclosure irradiance output partial_irradiance
End
```

Note: The radiative flux, irradiance, and radiosity of the partial enclosure, can be output by specifying the relevant command lines in the enclosure definition command block. However, unlike the enclosure variables corresponding to part of the surface mesh, partial enclosure variable outputs are written to global variables.

Banded Wavelength Model

When employing the *banded-wavelength enclosure model*, a reference must be made in the enclosure definition to a banded wavelength model command block. Additionally, band command blocks must also be defined in the enclosure definition with line commands specifying the emissivities on respective surfaces. For example, to define a three band wavelength model one would add the following command blocks and line commands:

```
Begin Aria Region My_Region
  ...

  Begin Enclosure Definition enc1
    add surface surface_1 surface_2
    ...

    # Define which banded wavelength model to use
    use banded wavelength model example_model

    # Specify the emissivity for each band on each surface
    begin band first
      emissivity = 0.4 on surface_1
      emissivity = 0.7 on surface_2
    end
    begin band second
      emissivity = 0.8 on surface_1
      emissivity = 0.7 on surface_2
    end
    begin band third
      emissivity = 0.8 on surface_1
      emissivity = 0.3 on surface_2
    end
  End

  # Define the banded wavelength model
  begin banded wavelength model example_model
    band first  = 0.0 3.0
```

(continues on next page)

(continued from previous page)

```
    band second = 3.0 5.0
    band third  = 5.0 1000.0
  end
  ...

  # Average across bands
  postprocess average of expression irradiance on surface_1 as G_avg

  # Per band average
  postprocess average of expression enclosure_1_irradiance_band0 on␣
↪surface_1 as G_band0_avg
  postprocess average of expression enclosure_1_irradiance_band1 on␣
↪surface_1 as G_band1_avg
  postprocess average of expression enclosure_1_irradiance_band2 on␣
↪surface_1 as G_band2_avg

  # Define enclosure facet field variables for output
  Begin Results Output Label encl_rad
    ...

    # facet field variables summed across bands
    face variables = radiosity as J
    face variables = rad_flux as q
    face variables = irradiance as G

    # band specific facet field variables
    face variables = enclosure_1_radiosity as encl_1_J
    face variables = enclosure_1_rad_flux as encl_1_q
    face variables = enclosure_1_irradiance as encl_1_G

    # specify face variables in 3D, edge variables in 2D
  End
End
```

Note: The emissivity must be specified on all surfaces for each band.

When using the banded wavelength model the radiative flux, irradiation and radiosity can be output on each band. Here, the respective face field variable have been prefixed with ENCLOSURE_EID, where EID is the enclosure index (the order index corresponding to where the enclosure definition appears in the input command file). In this case the values corresponding

to each band will be output. Additionally, per band expressions for the radiative flux, irradiation, and radiosity are also provided. As shown above, the expression names correspond to the face field name with the zero-based band index appended.

Other command references for the banded wavelength model can be found *here*.

Dash Enclosures

Instead of enumerating surfaces of the enclosure, the dash enclosure capability automatically generates detected enclosures while simultaneously remedying defects in the surface mesh of candidate enclosures. Dash enclosures are constructed of hybrid (quadrilateral/triangle) topology, where triangle subfacets are introduced as needed to define the enclosure cavity. This methodology is applicable to both merged (contiguous mesh) or unmerged (discontiguous mesh) geometries.

Beta Capability
For complex geometries, the automatic enclosure detection can struggle to generate fully closed (watertight) enclosures. Please check the rowsum error and visualize the automatically detected enclosures to ensure correctness.

As shown in the example input deck snippet below, the DASH algorithm can be applied universally to skin all blocks by including `skinned_blocks` or can be applied to a subset of the mesh by adding specific blocks to skin and or specific surfaces to include, either of which can also correspond to assemblies or mesh groups. The latter of which is useful when the problem only has energy equation types on a subset of the mesh.

```
Begin Aria Region My_Region
  ...

  # Define enclosures using dash
  Begin Enclosure Definition enc1
    use dash enclosures

    # for visualizing enclosures
    preprocess enclosures

    # include skinned surface from specific blocks plus a surface␣
↪assembly / mesh group
    add surface block_1 block_2 surfaces_to_include

    # alternatively, skin all blocks in the mesh
    # add surface skinned_blocks
```

(continues on next page)

(continued from previous page)

```
    # only solve dash enclosure with id = 1
    dash solve enclosures = 1
    ...
  End

  Begin Results Output Label encl_rad
    ...
    face variables = enclosure_list as encl_ids
  End

  ...
End
```

Oftentimes dash enclosures result in several enclosures being detected. By default a radiosity solve occurs for each detected enclosure. The `dash solve enclosures` line command can be added to restrict which enclosures should be solved by specifying a list of enclosure ids.

To determine enclosure ids in meshed models, the `preprocess enclosures` command can be used to compute integer face fields that can be output by requesting a face variable `enclosure_list` be output to the results file. In the example above, output for the `enclosure_list` is five integer face fields (e.g. `elist_1`, `elist_2`, etc.) as the DASH algorithm restricts each face to belong to at most five enclosures. The `elist_2` face field on skinned mesh parts contains enclosure facet values over a range of integers, but interest here lies in values > 0. Threshold or isovolume visualization of `elist_2` about integers >= 1 will reveal the surface geometry for Dash enclosure ENCL_INTEGER. For instances where subfacets have been introduced, the surface geometry will include the original (non-subfacetted) facets as well as the enclosure cavity.

Note: The enclosure id referenced in `dash solve enclosures` is zero based whereas the integers in the face fields are one based i.e., ENCL_ID = ENCL_INTEGER - 1.

For both continuous and discontiguously meshed models, the enclosure id can also be obtained using the `preprocess enclosures` command, and including either a rowsum or topology database name specification in the enclosure definition. Here a cavity database whose name contains the associated enclosure id will be written for each dash enclosure.

The `dash solve enclosures` line command can also be used to circumvent the restriction that all line commands in a given enclosure definition command block will apply to all resulting enclosures. For example, the enclosures can be split across separate enclosure definition blocks as follows:

```
Begin Aria Region My_Region
```

(continues on next page)

(continued from previous page)

```
  ...

  Begin Enclosure Definition enc0
    use dash enclosures

    # only solve dash enclosure with id = 0
    dash solve enclosures = 0

    add surface skinned_blocks
    Use Viewfactor Calculation vf_calc0
    Use Viewfactor Smoothing no_smooth0
    Use Radiosity Solver Rad_Solv0
    ...
  End

  Begin Enclosure Definition enc1
    use dash enclosures

    # only solve dash enclosure with id = 1
    dash solve enclosures = 1

    add surface skinned_blocks
    Use Viewfactor Calculation vf_calc1
    Use Viewfactor Smoothing no_smooth1
    Use Radiosity Solver Rad_Solv1
    ...
  End
  ...
End
```

Enclosures and Mesh Adaptivity

Enclosure radiation calculations can quickly become expensive when used with mesh adaptivity. If the mesh is adapted such that some of the faces included in the enclosure are modified, the entire viewfactor calculation for that enclosure must be re-done. If the mesh is adapted frequently, this can quickly make the simulation cost become intractable. One solution to this is to decouple the mesh used in enclosure radiation from the mesh used in the rest of the simulation. This can be done with shell blocks that mirror the geometry of the block surfaces participating in enclosure radiation. However, since the shell blocks are meshed independently of the volume blocks, adaptivity of the volume block will not impact the shell, avoiding a viewfactor recompute. The shell blocks can even be configured to have an independent adaptivity scheme applied to them so they can be adapted at a different frequency than the volume block (or not at all).

Aria provides the necessary machinery for computing enclosure radiation using shells and transferring temperatures and the resulting fluxes between shells and volume blocks using contact. There are two methods, one requiring the user to provide the shell blocks they wish to use and configure their own contact mechanics, and an automated method where specified surfaces are used to create shell blocks on the fly and default contact settings used. For user-provided shells, both the shell and an attached surface, oriented correctly for the enclosure calculation, must be included in the mesh file and a contact block must be specified to define how the shell surfaces will interact with the volume mesh.

```
Begin Aria Region My_Region
  ...

  Begin Enclosure Definition enc1
    add surface surface_1 surface_2 surface_1000
    ...

  End

  Begin Contact Definition surf_to_surf
    Contact Surface surf_1 Contains surface_3
    Contact Surface surf_2 Contains surface_1000
    ...

    Begin Enforcement enf1
      Enforcement for Energy = Shell_Enclosure Enclosure = enc1
    End
  End
End
```

In the above example, `surface_1000` is attached to a shell block and is being used instead of `surface_3` in the enclosure calculation. The `Shell_Enclosure` contact enforcement model will provide a contact energy flux on `surface_3` resulting from the solution of the radiosity equation and interpolated between the shell mesh and the surface mesh of `surface_3`. Additionally, the contact enforcement will automatically:

- Connect with the enclosure definition specified in the contact enforcement and make sure the same shell surface is included in each
- Invert the usual contact opposed normal check to allow `surface_1000` and `surface_3` to be oriented in the same direction and still be in contact
- Add a dummy (`XFER` only) energy equation to the shell `surface_1000` is connected to if it does not already exist
- Create a utility to interpolate temperature values from `surface_3` to `surface_1000`
- Create a utility to interpolate radiosity solution fields (`RAD_FLUX`, `IRRADIANCE`, and

RADIOSITY) from surface_1000 to surface_3

In this example, if surface_3 and surface_1000 were to have the same mesh, the problem results would be the same (although the path to solution could be different). In general, however, this method is useful as it allows surface_3 and surface_1000 to have different meshes, or to be refined on different schedules.

If the user wants to use default contact settings and a shell mesh that's copied from an existing mesh surface, a simplified syntax is available:

```
Begin Enclosure Definition enc1
  add surface surface_1 surface_2
  use skinned shells on surface surface_3
  ...
  emissivity = 0.6 on surface_3
End
```

In this case, a shell and attached surface mesh part will be created that's a copy of surface_3, and the contact definition will be created automatically. The calculation will then proceed as if the created shell had been included in the input mesh. Emissivity values can even be specified as part of the enclosure definition for surface_3, as they normally can be, and these definitions will carry through to the created shell surface. The created shell block parts use the naming scheme [ENCLOSURE NAME]_SHELL_FOR_[SURFACE NAME] and the surface parts are named [ENCLOSURE NAME]_SHELL_FOR_[SURFACE NAME]_SURFACE, although for most purposes the user should not need to refer to the created shells and surfaces directly as all solution quantities are interpolated back to the originating surface for post-processing. The main exception to this is if the user wants to define an adaptivity scheme on the created shell.

The primary limitation of the automatic shell generation for enclosure calculations is that it cannot be used with block skinning. That is, you cannot automatically create a shell for a surface that itself is automatically created from block skinning.

Using shell-lined enclosures in conjunction with the Viewfactor Update = only_if_geometry_changed option provides a method to control the number of viewfactor calculations performed for problems using both enclosure radiation and adaptivity. Note that since the transfer between shell and block surfaces is done using contact, a new contact search must be performed whenever the block surface is changed due to adaptivity. However, the cost of a contact search is generally orders of magnitude less than a viewfactor computation. Users are cautioned, however, that doing no or infrequent mesh adaptions on enclosure surfaces can result in the enclosure mesh becoming the dominant source of solution discretization error. Mesh convergence studies will need to take this into account and an appropriate mesh used for the enclosure to control error.

The input file snippet shown below shows how everything can be used together.

```
Begin adaptivity options my_block_adapt
  begin uniform marker my_block_marker
```

(continues on next page)

(continued from previous page)

```
    maximum refinement level = 5
    location = block_1
  end
End

Begin adaptivity options my_shell_adapt
  begin uniform marker my_shell_marker
    maximum refinement level = 3
    location = ENC1_SHELL_FOR_SURFACE_3
  end
End

Begin Procedure MyProcedure
  Begin Solution Control Description
  ...

    Begin System Main
      ...

      Begin Transient TimeBlock
        # Adapt the enclosure shell mesh 10x less frequently than the␣
↪bulk mesh
        Adapt mesh for AriaRegion using my_block_adapt when "transient_
↪timeblock.CURRENT_STEP % 10 == 0"
        Adapt mesh for AriaRegion using my_shell_adapt when "transient_
↪timeblock.CURRENT_STEP % 100 == 0"
        Advance AriaRegion
      End
    End
  End

  Begin Aria Region AriaRegion
    ...

    Begin Enclosure Definition enc1
      add surface surface_1 surface_2
      # Generate enclosure shells for surface_3
      use skinned shells on surface surface_3
      # Only update viewfactors if enclosure mesh has changed
      Viewfactor Update = only_if_geometry_changed
      ...
```

(continues on next page)

(continued from previous page)

```
    End
  End
End
```

SPn Radiation

Beta Capability
SP_N is not well tested and should not be considered a production capability.

Below is an example input deck with relevant command blocks for the *SPn equations* coupled monolithically to the energy equation through a radiative heating source and radiative heat flux. Note several command lines and blocks have been left out to focus on the incremental change needed to add the SP_N equations (with relevant boundary conditions and initial conditions) to a model.

```
# Use aprepro to define the SPn order as it will need to be referenced
# in multiple locations in a consistent fashion.
#{SPN_ORDER=3}

Begin Sierra myJob
  ...

  Begin Global Constants
    Stefan Boltzmann Constant = 5.67e-8
  End

  # define absorbtion, scatter, emissivity models for the block
  Begin Aria Material my_pmr_material
    ...
    absorption cross section = constant abs = 0.5
    scattering cross section = constant scat = 0.1
    emissivity = constant e = 1.0
  End
  ...

  Begin Procedure My_Aria_Procedure
    ...
    Begin Aria Region My_Region
      ...
```

(continues on next page)

(continued from previous page)

```
      # Define steady heat conduction
      EQ energy for temperature on block_1 using Q1 with diff src
      BC dirichlet for temperature on my_dirichlet_surfaces = constant␣
↪value = 1000

      # Couple to SPn eqns via radiative heat source
      source for energy on block_1 = rte_sp order = {SPN_ORDER}

      # Couple to SPn eqns via radiative flux
      BC flux for energy on my_rad_flux_surfaces = RTE_SP order = {SPN_
↪ORDER}

      # Define SPn radiation equation set of order N resulting in (N + 1)␣
↪/ 2 angular intensity DOFs
      EQ sp {SPN_ORDER} for angular_intensity on block_1 using Q1

      # Apply Mark BCs for the SPn equation(s)
      BC Flux for sp on all_surfaces = mixed

      IC for temperature on block_1 = constant value = 1000.0

      # Define SPn ICs (initial guess to steady problem)
      IC for angular_intensity_0 on block_1 = constant value = 10420.
↪116122115
      IC for angular_intensity_1 on block_1 = constant value = 10420.
↪116122115

      # Define SPn nodal variables for output
      Begin Results Output Label Spn
        ...
        nodal variables = rte_scalar_flux as G
        nodal variables = rte_radiative_flux as q
        nodal variables = SPn_flux_0 as S0
        nodal variables = SPn_flux_1 as S1
        nodal variables = solution->angular_Intensity_0 as I0
        nodal variables = solution->angular_Intensity_1 as I1
      End
    End
  End
End Sierra myJob
```

MPMD Radiation

This chapter outlines the use of Multiple-Program-Multiple-Data (MPMD) coupling to perform radiation solves with Sierra/PMR. MPMD coupling allows the radiation solve to occur on an entirely different set of processors using an executable other than Aria. See *PMR - Radiation* for a complete example of an MPMD radiation problem.

For information on launching an MPMD simulation, see *Running MPMD Jobs*. Note that for the case of the Aria-PMR coupling, Aria must be the `mpmd_master`, so an example launch line would be

```
$ mpirun -np 100 aria --mpmd_master --mpi_color 9999 \
    -i input.i : -np 100 pmr -i pmr.i
```

Aria Input Deck

Enabling MPMD radiation coupling in the Aria input deck requires the addition of an MPMD Coupling block to the Aria region. See the *command reference* for details on all available lines.

```
# Scope: Sierra > Procedure > Aria Region
Begin MPMD Coupling
  Coupling Type is Radiation
  Begin Radiation Coupling
    Iterations per radiation solve is #
    #    -or-
    Time steps per radiation solve is #
  End Radiation Coupling
End MPMD Coupling
```

The only option to specify is the frequency with which the radiation transport equation solve should occur with respect to the Aria equation solves. Specifying Iterations per radiation solve will set the RTE to be solved every specified number of Aria non-linear iterations. Specifying Time steps per radiation solve will set the RTE to be solved every specified number of time steps. Both options cannot be specified simultaneously and neither are required. The default behavior is to run the RTE solve every iteration.

The mesh parts that the *RTE* should be solved on are specified by including them as equation descriptions lines in the input file as described in *Equation Specification*.

PMR Input Deck

The PMR input deck syntax is described in the current PMR user manual. It is YAML formatted and can be quite terse for most cases. The common parameters to set in the PMR input file are the mesh file to use, the quadrature order, solution method, and linear solver settings. The RTE equation does not have a time term, so no transient information is needed for PMR. The PMR code will terminate automatically when Aria completes its execution.

Ray Tracing

Aria includes a ray tracing capability that can be used to model the effect of a focused radiation source on a target. Ray tracing works by computing the shortest intersection of a ray and mesh facets in the domain. When an intersection is detected, the ray will interact with the surface, with a portion of its energy deposited to drive a flux boundary condition, and a portion reflected away to potentially interact with other facets. For certain problems, ray tracing can offer performance advantages over *enclosure radiation* and *PMR using MPMD*. Specifically, when radiation transport is dominated by focused sources of radiation (like a laser), re-radiation between mesh surfaces can be neglected, and participating media effects can be neglected.

To enable ray tracing, first a ray source must be defined at the equation system level. The ray source defines the origin and distribution of rays that will be traced through the domain. The ray source also defines the surface properties and absorption model that will be used to compute the energy deposition that occurs when a ray interacts with a facet as well as some options that govern the behavior of the ray tracer.

```
Begin Raytrace Source Ray_Model
  Type = Circular_Gaussian
  Path Function = path
  Direction Function = direction
  Power Function = laserPower
  Radius = 0.1
  Exponential Factor = 2.0
  Maximum Intersections = 10
  Minimum Power Ratio = 0.001
  Discretization = Radial
  Number of Radial Segments = 50
  Number of Circular Segments = 5
  Absorption Model = specular
  Absorption expression = emissivity1
  Parallel Type = scene
End Raytrace Source Ray_model

#The ray source can then be used in a flux BC
BC Flux for Energy on surface_1 = Ray_Trace Ray_Source = Ray_Model
```

The TYPE specifies the distribution of the rays to be generated. Only circular distributions currently exist and can have either a Gaussian flux distribution ('CIRCULAR_GAUSSIAN' option) or constant ('CIRCULAR' option). The PATH FUNCTION, DIRECTION FUNCTION, and POWER FUNCTION specify *Sierra user functions* that govern the center location, initial direction, and total power of all the rays as a function of time. PATH FUNCTION and DIRECTION FUNCTION expect multi-component functions with X, Y, and Z components.

RADIUS and EXPONENTIAL FACTOR govern the Gaussian flux distributions as given by:

$$q_{gaussian} = \frac{(ExpFactor)(P)}{\pi Radius^2} \exp\left(-ExpFactor\left(\frac{d^2}{Radius^2}\right)\right)$$

where d is the distance to the origin of the circle and P is the current value of power given by the POWER FUNCTION. The constant flux distribution only requires specifying RADIUS and is equal to:

$$q_{constant} = \frac{P}{\pi Radius^2}$$

The constant flux distribution also allows the specification of the flux directly using the FLUX FUNCTION command instead of the POWER FUNCTION.

MAXIMUM INTERSCTIONS and MINIMUM POWER RATIO determine cutoff criteria for when rays should no longer be traced.

The DISCRETIZATION command controls how the flux distribution is discretized into rays. Valid options are 'RADIAL' and 'MONTE_CARLO'. For the MONTE_CARLO option, rays origins are randomly generated within a circle of radius RADIUS for constant flux and anywhere in 2D space for Gaussian flux. For constant flux, rays will be uniformly distributed within the circle, whereas for Gaussian flux the distribution will be Gaussian in distance from the center as specified by PATH FUNCTION, meaning more rays will be generated closer to the center. For the 'MONTE_CARLO' discretization, the NUMBER OF RAYS option must be given to set the number of random rays to generate. The initial power of each ray will then be the total power specified by the POWER FUNCTION divided by NUMBER OF RAYS.

For the 'RADIAL' discretization, a deterministic method is used to generate rays. A circle of radius RADIUS for the constant flux distribution and $2 * RADIUS$ for the Gaussian flux distribution is split into a series of segments as controlled by the NUMBER OF RADIAL SEGMENTS and NUMBER OF CIRCULAR SEGMENTS options. The number of radial segments determines the number of rings the circle is broken into. The number of radial segments determines the number of angular segments in the innermost ring. The number of angular segments per ring increases as the ring radius increases to the area of each segment is approximately equal. The total number of rays generated will be $NumberOfRadialSegments^2 * NumberOfCircularSegments$. The flux distribution is integrated for each segment to give a power and a ray generated from the center of the segment with that power. For the Gaussian distribution, the outermost ring is assumed to extend to infinity for the purposes of integrating the flux, but ends at $2 * RADIUS$ for purposes of determining the ray origin.

The ABSORPTION MODEL option controls how rays will interact with mesh facets they intersect with. The options are 'SPECULAR' and 'FRESNEL'. For the specular model, a fraction of the

ray's power equal to ϵ will be absorbed and the remaining $1 - \epsilon$ will be reflected to possibly interact with another facet. ϵ is given by the expression provided in the `ABSORPTION EXPRESSION` option, allowing a user to define their own *user expression* if they wish. If this option is omitted, the *EMISSIVITY* expression will be used.

The 'FRESNEL' model calculates the fraction of absorbed ray power based on the angle of incidence of the ray. This is given by:

$$\alpha = 1 - \frac{1}{2}\left(\frac{1 + (1 - \epsilon cos\phi)^2}{1 + (1 + \epsilon cos\phi)^2} + \frac{\epsilon^2 - 2\epsilon cos\phi + 2cos^2\phi}{\epsilon^2 + 2\epsilon cos\phi + 2cos^2\phi}\right)$$

α is the absorbed power fraction, ϕ is the angle between the facet normal and the incoming ray, and ϵ is again provided by the `ABSORPTION EXPRESSION` option, which can be *user expression* or will default to the *REFRACTIVE_INDEX_RATIO* expression if omitted.

The `PARALLEL TYPE` option controls how ray trace calculations will be performed on parallel machines. With 'SCENE' parallelism, all processors have a copy of all the rays, which they intersect with their local piece of the distributed mesh. A communication is then done to resolve the shortest path when a ray intersection is found on multiple processors. With 'RAY' parallelism, all processors have a copy of the facets that the rays could possibly intersect with and the rays to be traced are distributed across processors. 'RAY' parallelism is faster, but incurs larger memory overhead, which can cause problems on larger meshes. 'SCENE' parallelism is slightly less performant, but requires only minimal additional memory.

Finally, the defined ray source can be used to define a boundary condition on a surface using the `RAY_TRACE` Flux BC model. Defining a `RAY_TRACE` BC on a mesh surface will automatically add the facets in that surface to the list of possible facets to intersect with for the specified ray source. The `RAY_TRACE` BC can optionally include black body emission from the surface using the `EMISSION=TRUE` option.

Banded Radiation

The ray tracer also supports banded emissivity models by allowing a user to specify a flux/power per band and an absorption expression per band.

```
#Define banded wavelength model
begin banded wavelength model test_band
  band first  = 0.0 3.0
  band second = 3.0 5.0
  band third  = 5.0 1000.0
end banded wavelength model test_band

Begin Raytrace Source Ray_Model
  Type = Circular_Gaussian
  Path Function = path
```

(continues on next page)

(continued from previous page)

```
  Direction Function = direction
  Radius = 0.1
  Exponential Factor = 2.0
  Maximum Intersections = 10
  Minimum Power Ratio = 0.001
  Discretization = Radial
  Number of Radial Segments = 50
  Number of Circular Segments = 5
  Banded Wavelength Model = test_band
  Power Function for band first = laserPower1
  Power Function for band second = laserPower2
  Power Function for band third = laserPower3
  Absorption expression for band first = emissivity1
  Absorption expression for band second = emissivity2
  Absorption expression for band third = emissivity3
  Parallel Type = scene
End Raytrace Source Ray_model

#The ray source can then be used in a flux BC
BC Flux for Energy on surface_1 = Ray_Trace Ray_Source = Ray_Model␣
↪Emission = True
```

In this case, the `ABSORPTION MODEL` option is omitted as only specular reflection is supported for banded cases. A *banded wavelength model* must also be specified in the equation system scope and the name provided to the ray source using the `BANDED WAVELENGTH MODEL` option. If the emission option is specified in the Flux BC command for a banded wavelength ray source, a band-averaged emissivity will be calculated.

Contact

This section discusses how to define a contact boundary between non-contiguously meshed portions of the domain. For a detailed description of the definition of contact and the mathematical basis of the available enforcement models, the reader is referred to the *theory section*. A comprehensive list of command line options related to contact definitions can be found in the *command reference*.

Most common cases model contact between surfaces of standard volume elements (2D quadrilateral, 2D triangle, 3D tetrahedron or 3D hexahedron). When considering contact of shell elements with nonconformal interfaces, it is assumed that the shell nodes are coincident with nodes of a standard volume element on one side of the contact under consideration.

Warning: Shell support is limited in nonconformal contact. Specifically, contact along the edge of a shell element is not supported, i.e., a shell element that is perpendicular to another surface does not transmit a flux through the singular axis of the shell.

A contact definition has two main parts, 1) specifying which parts of the mesh are in contact (interactions), and 2) specifying what condition(s) should be applied on the portion of those mesh parts that are in contact (enforcement). An example contact definition is provided below:

```
# Scope: Sierra > Procedure > Aria Region
Begin Contact Definition surf_to_surf

  # Interaction: Define which entities are in contact, all pairs of␣
↪contact surfaces will be checked for
  # contact. By default if no specific "Begin Interaction" blocks are␣
↪present then Aria will check for contact
  # between all contact surfaces. In this example contact will be␣
↪applied between [surface_3, surface_4],
  # [surface_3, surface_5], and [surface_4, surface_5]
  Contact Surface surf_1 Contains surface_3
  Contact Surface surf_2 Contains surface_4
  Contact Surface surf_3 Contains surface_5

  # Enforcement: How to treat this contact
  Begin Enforcement enf1
    Enforcement for [EQ] = [MODEL] [MODEL ARGS...]
  End

End
```

Interactions

The first step of defining contact interactions is to specify which entities are potentially in contact. In the simplest cases this is done by adding the surfaces of interested to a `Contact Surface` (as is demonstrated in the first example above). Note that *assemblies* can be used to simplify the specification of the contacting surfaces.

Alternatively, contacting surfaces can be automatically be found by "skinning" blocks, which automatically creates a surface containing the exterior faces of the blocks, and running contact searches on those surfaces. This can be done either for specific blocks:

```
# Scope: Sierra > Procedure > Aria Region
begin contact definition mixed_skinning
```

(continues on next page)

(continued from previous page)

```
  #Interaction is a mix of surfaces and skinned blocks
  Contact Surface surf_1 Contains surface_1 block_3 surface_2
  Contact Surface surf_2 Contains surface_3 block_4 surface_5

  Begin Search Options
    # tol can be calculated directly in input
    normal tolerance = {0.1 * lChar}
  End

  begin enforcement enf_1
    Enforcement for Energy = tied_temperature
  end
end
```

Or for complex geometries with many contacting surfaces that cannot be easily defined/specified, the *skin all blocks* command can be used to apply contact between the exterior surfaces of every block in the mesh as in this example:

```
# Scope: Sierra > Procedure > Aria Region
begin contact definition all_to_all

  Begin Search Options
    normal tolerance = 0.01
  End

  # All skinned blocks participate
  skin all blocks = on

  begin enforcement enf_1
    Enforcement for Energy = tied_temperature
  end
end
```

Finally for more complex subsets of skinned surfaces, one can define a *Skinned Surface* where the contact is to be defined e.g.

```
# Scope: Sierra > Procedure > Aria Region
begin skinned surface first_surf
  touching blocks = block_1
  excluding surface = surface_to_ignore
end
begin skinned surface second_surf
```

(continues on next page)

(continued from previous page)

```
  touching blocks = block_2
  excluding surface = surface_to_ignore
end

begin contact definition mixed_skinning

  #Interaction is a mix of surfaces and skinned blocks
  Contact Surface surf_1 Contains first_surf
  Contact Surface surf_2 Contains second_surf

  Begin Search Options
    # tol can be calculated directly in input
    normal tolerance = {0.1 * lChar}
  End

  begin enforcement enf_1
    Enforcement for Energy = tied_temperature
  end
end
```

In some cases it can be useful to limit the contact to only be between specific pairs of contact surfaces. This can be accomplished by providing specific **Begin** `Interaction` blocks in the contact definition as in this example:

```
# Scope: Sierra > Procedure > Aria Region
Begin Contact Definition surf_to_surf

   # Interaction: Define which entities are in contact, all pairs of␣
↪contact surfaces will be checked for
   # contact.  In this example contact will only be applied between␣
↪[surface_3, surface_4],
   # and [surface_4, surface_5]
   Contact Surface surf_1 Contains surface_3
   Contact Surface surf_2 Contains surface_4
   Contact Surface surf_3 Contains surface_5

   Begin Interaction s1s2
     surfaces = surf_1 surf_2
   End
   Begin Interaction s2s3
     surfaces = surf_2 surf_3
   End
```

(continues on next page)

(continued from previous page)

```
  # Enforcement: How to treat this contact
  Begin Enforcement enf1
    Enforcement for [EQ] = [MODEL] [MODEL ARGS...]
  End

End
```

Self contact (a condition where part of a contact surface is in contact with itself) can occur in some cases. Common examples of this include:

- Non-convex geometries
- Large or complex geometries
- Geometries where two volumes are meshed separately and added to the same block

In these cases, it is necessary to activate self contact in the *interaction defaults* by specifying

```
begin contact definition assembly_contact
  ...
  begin interaction defaults
    self contact = on
  end
end
```

Search and Contact Tolerance Calculation

The enforcement selected only acts upon surfaces that are determined to be in contact based on geometric proximity. Whether or not surfaces are in contact is determined via a normal tolerance that is either 1) User specified within the search options or interaction blocks of the contact definition (see *simulation setup*), or 2) set via an automatic tolerancing operation based on the mesh size. This normal tolerance is used on a per-integration-point projection to the closest surface, as shown in Fig. 4.15.

In more detail: For a given surface that may be in contact, the enforcement strategy is expressed in terms of a surface integral on both surfaces (surface a and surface b in Fig. 4.15). For evaluating the enforcement kernel on surface a, the surface facets are iterated, as are the integration points within each facet. The closest surface is found, as defined by the smallest distance, L, from the integration point to the surface intersection along the normal direction (L in Fig. 4.15). If $L < tol$, where tol is the contact normal tolerance, then that integration point is in contact and the enforcement kernel is evaluated. If no global or interaction specific contact normal tolerance is specified, the automatically calculated contact normal tolerance is used, which is calculated as the minimum elemental characteristic length over all surface elements in a given contact surface. The elemental characteristic length is defined as the minimum length between any two nodes on the surface element.

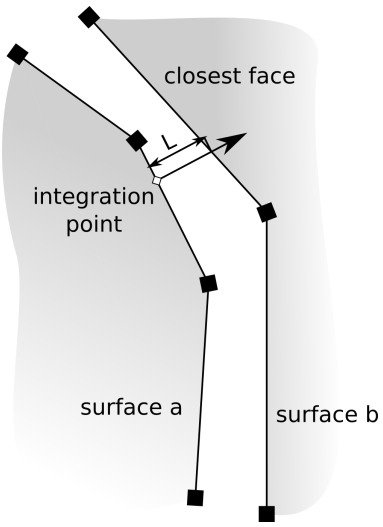


Fig. 4.15: Calculation of contact distance per integration point.

Note: If the smallest distance that might reasonably be observed in the gap is known, it is suggested that this value be expressly stated. This tolerance can be specified either *per interaction* or for all interactions in *search options*.

Note that if you want to override the tolerance for a specific interaction, but still have contact between all contact surfaces you need to set `General Contact = on` in the interaction defaults block as in this example:

```
# Scope: Sierra > Procedure > Aria Region
Begin Contact Definition surf_to_surf

  # Interaction: Define which entities are in contact, all pairs of␣
↪contact surfaces will be checked for
  # contact. In this example contact will be applied between [surface_3,␣
↪surface_4] and [surface_3, surface_5]
  # using the default mesh-based normal tolerance, while the contact␣
↪between [surface_4, surface_5] will
  # use the specified normal tolerance of 1.e-3
  Contact Surface surf_1 Contains surface_3
  Contact Surface surf_2 Contains surface_4
  Contact Surface surf_3 Contains surface_5

  Begin Interaction Defaults
    General Contact = on
  End
  Begin Interaction s2s3
```

(continues on next page)

(continued from previous page)

```
    surfaces = surf_2 surf_3
    normal tolerance = 1.e-3
  End

  # Enforcement: How to treat this contact
  Begin Enforcement enf1
    Enforcement for [EQ] = [MODEL] [MODEL ARGS...]
  End
End
```

Dynamic Contact

By default, the contact search is updated only when mesh modification occurs. In problems where the mesh is deforming with time (e.g. with `EQ MESH` or coupling to Adagio), it can be necessary to also rerun the search as interfaces move with respect to each other. This is accomplished by specifying an *update frequency* `Update Search Every <N> Steps`. This will cause the search to update not only when mesh modification occurs, but ever `N` time-steps as well.

Enforcement

The full list of contact enforcement models available for each equation in Aria is available in the *command reference*.

For thermal contact problems there are two primary cases of interest that we discuss here: zero contact resistance across a mesh interface, and imperfect contact between materials on either side of the interface. For cases where there is no contact resistance, i.e., the conserved quantity field is contiguous, the team recommends using the `TIED_CONTACT` enforcement strategy. For cases with imperfect contact that can be modeled according to a contact resistance scheme, the team recommends using the `GAP_CONDUCTANCE` enforcement strategy. These strategies have the highest pedigree of code testing, verification, and validation. In particular the `GAP_CONDUCTANCE` enforcement model is more robust than the `CONTACT_RESISTANCE` model for large conductance coefficient values. See section 4.3 of [20] for more details.

Custom enforcement models can also be applied using the string function model. For example, the following two contact resistance enforcement models are equivalent.

```
# Scope: Sierra > Procedure > Aria Region
Begin contact definition my_def
  ...
  begin enforcement enf_1
    Enforcement for Energy = contact_resistance
    gap conductance coefficient = constant value = 500
```

(continues on next page)

(continued from previous page)

```
  end enforcement
End
```

```
# Scope: Sierra > Procedure > Aria Region
Begin contact definition my_def
  ...
  begin enforcement enf_1
    Enforcement for Energy = scalar_string_function f = "500*(temperature.
↪current - temperature.opposite)"
  end enforcement
End
```

When using a string function in a contact enforcement model the "current" and "opposite" suffixes can be used to refer to properties or values on either side of the interface. Custom properties can be applied to either surface using a surface material, or pulled from the volume material on either side of the interface. As with other string functions, global variables can also be used with the "global" suffix.

Contact Materials

For some of the *Enforcement*s listed above, material properties must be defined at the interface. The most common example of this is providing the `GAP CONDUCTANCE COEFFICIENT` for resistance-based enforcement models. These material properties can be defined for a contact interface either in the surface material properties or per-enforcement in the enforcement block. As an example in the latter case

```
# Scope: Sierra > Procedure > Aria Region
Begin contact definition my_def
  ...
  begin enforcement enf_1
    Enforcement for Energy = contact_resistance

    gap conductance coefficient = constant value = 1e6
  end enforcement
End
```

Note: If the `GAP CONDUCTANCE COEFFICIENT` is not defined on a contact surface or its corresponding block material, the value on the opposing surface will be taken.

Warning: If a required contact material property cannot be found, a *missing expression error* will be thrown.

It is also possible to specify different conductance coefficients per-enforcement. This is done by appending the `EQUATIONS` to apply a given model to at the end of the material line definition as is demonstrated below

```
# Scope: Sierra > Procedure > Aria Region
Begin contact definition my_def
  ...
  begin enforcement enf_1
    Enforcement for Energy = contact_resistance
    Enforcement For Current = gap_conductance

    gap conductance coefficient = constant value = 1e6 equations = Current

    contact surface pressure = either_or source=contact_pressure
    gap conductance coefficient = user_function name=gap_cond \$
      x=contact_surface_pressure x_multiplier = -1.0 equations = Energy
  end enforcement
End
```

The full list of contact material models is available in the *command reference*.

Shell Lofting

Warning: Shell lofting is a work-in-progress and is not thoroughly tested. It is not recommended for production use.

When dealing with shells in contact, the displacement of the shell coordinates for the purposes of contact is based off of the shell lofting factor. The default lofting factor $f = 1$ results in a shell that is "lofted", i.e., displaced, by the shell thickness in the direction of the shell normal. A lofting factor of 0.5 results in a shell whose midplane is bisected by the defining surface, and a lofting factor of -1 results in a shell surface displaced opposite of the shell normal.

Periodic Contact

In problems that are periodic in space, external surfaces may corresponds to periodicity planes. It is often difficult to make the meshes on these surfaces match exactly, such that a *periodic BC* can be applied. As a more general alternative, the two surfaces can be tied via contact. In this case, periodicity is specified to describe how the two surfaces match up. Periodicity commands will have the general form

```
Begin Contact Definition contact
  ...

  periodicity for [CONTACT SURFACE] = [MODEL] [model args...]
End
```

where the `CONTACT SURFACE` is one defined with a *contact surface* command. The following `MODEL`s are supported

ALIGN_SURFACE

This model determines the centroid and average surface normal of the specified `CONTACT SURFACE`, and translates/rotates it to align with those of a specified destination surface. If the exact offset is not know, this can be the most convenient model to use. Note that the two surface *do not* need to have the same topologies.

Warning: Care must be taken when this model is used for surfaces whose geometries do not match, or where the average surface normal is zero.

RECTANGULAR

The specified `CONTACT SURFACE` is translated $\boldsymbol{p}' = \boldsymbol{p} + \boldsymbol{\Delta}$ by offset $\boldsymbol{\Delta}$ before performing contact search.

CYLINDRICAL

The specified `CONTACT SURFACE` is rotated around the given axis and origin before performing contact search.

MAPPED

With this model, points $\boldsymbol{p}$ on specified surface are mapped via string functions $\boldsymbol{p}' = f(\boldsymbol{p})$ before performing contact search.

See the *command reference* for more details on specifying periodicity.

Offset Quantities

In some problems, a specific jump across a contact interface is required. This can occur either between adjacent blocks or in a *Periodic Contact* setting. In either case, `OFFSET` material properties can be defined to impose the offset

```
# Scope: Sierra > Procedure > Aria Region
Begin Contact Definition res1
  Contact Surface surf_1 Contains surface_1
  Contact Surface surf_2 Contains surface_2

  Begin Enforcement enf_1
    Enforcement for Energy = contact_resistance
    gap conductance coefficient = constant value = 5e5
    offset temperature = constant offset = 10.0 surface = surface_2
  End Enforcement enf_1
  ...
End Contact Definition res1
```

In *Periodic Contact*, offsets can be used flip vector values in vector enforcements. For example, consider the following mesh deformation problem

```
          |
          v
     +-----------+
     |     t     |
<--- | l       r | --->
     |     b     |
     +-----------+
```

The velocity on one side can be `reflected` so that the body will expand symmetrically

```
# Scope: Sierra > Procedure > Aria Region
Begin Contact Definition res1
  Contact Surface surf_1 Contains left
  Contact Surface surf_2 Contains right

  periodicity for surf_2 = align_surface destination = surf_1

  Begin Enforcement enf_1
    Enforcement For Mesh = Tied_Mesh_Displacements
    offset mesh velocity = reflected normal = 1.0 0.0 0.0 surface = left
  End Enforcement enf_1
  ...
```

(continues on next page)

(continued from previous page)

```
End Contact Definition res1
...
BC DIRICHLET for Mesh_Displacements_z on top = scalar_string_function f =
 ↪"-0.1*t"
BC DIRICHLET for Mesh_Displacements_z on bottom = CONSTANT Value = 0
```

See the *command reference* for more details on specifying offsets.

Warning: When specifying offset on a surface, one must take care to ensure the problem does not become ill-posed. For example with the geometry demonstrated above, the following contact definition is ill-posed if a constant dirichlet is applied

```
# Scope: Sierra > Procedure > Aria Region
Begin Contact Definition res1
  Contact Surface surf_1 Contains left
  Contact Surface surf_2 Contains right

  periodicity for surf_2 = align_surface destination = surf_1

  Begin Enforcement enf_1
    Enforcement for Energy = contact_resistance
    gap conductance coefficient = constant value = 5e5
    # temp_offset = {temp_offset = 10.0}
    offset temperature = constant offset = {temp_offset}␣
 ↪surface = right
  End Enforcement enf_1
  ...
End Contact Definition res1
...
#BC DIRICHLET for temperature on bottom = constant value = 300
 ↪#ill-posed
BC DIRICHLET for temperature on bottom = scalar_string_
 ↪function \$
  f = "300 + x/{x_len}*{temp_offset}" #good
```

With a constant dirichlet BC on `bottom`, the `OFFSET_TEMPERATURE` cannot change at the shared node and the problem will not converge.

Contact Coverage

When two surfaces are in contact, there is a coverage term calculated on these faces that is 1 on uncovered integration points (those not in contact) and 0 on face integration points that are in contact. If you have defined other flux boundary conditions on the contact surfaces they are automatically scaled by this mask, in order to remove their contributions from the parts of the interface that are in contact. For example, if you have a large surface that is only partially in contact like Sideset 1 in Fig. 4.16, you may want to apply a separate convective boundary condition on the exposed parts of that face. You can do so with the usual boundary condition commands and the portion of Sideset 1 that is in contact with Sideset 2 will not have any convective flux terms.

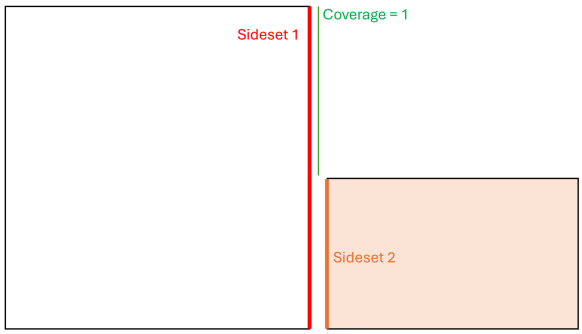


Fig. 4.16: Contact interface and coverage mask

If you want to ignore the coverage mask and apply a boundary condition over all the parts of Sideset 1 in the example above, you can use the `BC no_coverage_flux` command instead of `BC flux` to apply the boundary condition.

Warning: The coverage mask is applied to *all* flux boundary conditions, even those for DOFs that do not have a contact constraint

You can use the `COVERAGE` expression in postprocessors, but be aware that it will not be present unless there is an existing flux boundary condition on the surface. For example, in the example above you could use the following to postprocess the area on Sideset 1 that is in contact or not in contact.

```
# If needed, create a trivial flux BC so "coverage" is defined
BC Flux for Energy on sideset_1 = constant value = 0

Postprocess integral of expression coverage on sideset_1 as Cov1
Postprocess integral of function "1" on sideset_1 as Area1
Postprocess global_function "Area1-Cov1" as ContactArea1
```

If you use an expression postprocessor — for example to calculate the integrated flux from

convection on Sideset 1 in the example above — it will not include the coverage mask so it will include contributions in the contact area that were not actually applied. If the applied BC is a `no_coverage_flux` boundary condition this will be consistent, but not for a regular `flux` BC.

```
# Will match a "BC no_coverage_flux" but not "BC flux"
Postprocess integral of expression energy_flux on surface_1 as conv_flux

# Will match a "BC flux" but not "BC no_coverage_flux"
Postprocess integral of function "coverage*energy_flux" on surface_1 as␣
↪conv_flux
```

Debugging Contact

```
Begin Contact Definition contact
  ...
  visualize contact
  output rule = summary
End
...
Begin Results Output Label myOutput
  ...
  face variables = thermal_contact_status
End
```

While setting up a simulation, it can be helpful to receive debugging information about the contact search performed. This is controlled by specifying the `Output Rule` command line to the `Contact Definition`. Setting this to anything other than NONE will populate the log file with additional information that can be used in debugging contact. Along with enabling additional output printed to the log, it can be helpful to visualize where contact has been identified. This is enabled by adding the `Visualize Contact` command line to the `Contact Definition`. This command will populate a field called `thermal_contact_status` which can be added to the result database (see *Output Reference*) and visualized using e.g. Paraview. Surface facets will be marked by the percentage of their IPs where contact constraints are applied - 0 meaning no IPs and 1 meaning all IPs.

Warning: Use of this option is primarily for debugging purposes as the Results Output and History file will contain a skinned version of the contact element blocks. This may constitute a large amount of information for discontiguously meshed models.

Warning: Using contact visualization with multiple contact definitions on the same surface can result in duplicating contact contributions. Additionally, toggling each contact definition independently can result in errors in contact visualization.

Bulk Nodes

For some engineering problems the variation of a field variable over the problem domain can be roughly represented by some average behavior. In these cases it may be more appropriate to understand the averaged behavior rather than to resolve the behavior in fine detail. When numerically modeling these types of problems, it is convenient to devise lumped parameter models that simulate this averaged behavior.

One such modeling strategy, a bulk node model, is included in Aria to capture bulk fluid behavior without regard to spatial resolution. In this document the terms "bulk fluid element" and "bulk node" are often used interchangeably.

A bulk node defines a single new degree-of-freedom in your problem (often temperature) and sets up a conservation equation on that single node. The node can then be linked to the rest of the domain through boundary conditions coupled between sidesets on the mesh and the bulk node.

A *general conservation equation* containing mass and source terms for the bulk node degrees of freedom is included in the fully finite element model system of equations.

Some care is needed when selecting the heat transfer coefficient for a bulk node convection boundary condition. For example if modeling the fluid between two parallel plates via a bulk node, using a convection coefficient appropriate for the convective heat transfer between two plates would under-predict the heat transfer because that convection coefficient already accounts for the effect of the intervening fluid.

Example usage of the bulk node capability is outlined below, and a full command reference is available *here*. A bulk node requires a material model definition (a `Begin Aria Material` block at the Sierra level) as well as the `Begin bulk fluid element` block itself.

```
Begin Sierra myJob
  .
  Begin Aria Material a_bulk_volume
    # properties for bulk volume
  End
  .
  Begin Procedure My_Aria_Procedure
    .
    Begin Aria Region My_Region
      .
      Begin bulk fluid element bulk_volume_name
        material = a_bulk_volume
```

(continues on next page)

(continued from previous page)

```
        bulk EQ Energy for temperature using P0 with mass
        .
      End
      .
    End
    .
  End
  .
End Sierra myJob
```

A few example bulk node definitions are included below, showing how to set up the bulk node with a few standard configurations.

A simple thermal bulk node with a user-specified volume could be specified with:

```
Begin bulk fluid element bulk_volume_name
  material = a_bulk_volume
  bulk EQ Energy for temperature using P0 with mass
  initial temperature = 300
  bulk element volume = constant v = 0.01
End
```

On the other hand, the volume can be computed automatically by Aria based on the enclosing surfaces. The list of enclosing surfaces is determined by the boundary conditions that you link to this bulk node.

```
Begin bulk fluid element bulk_volume_name
  material = a_bulk_volume
  bulk EQ Energy for temperature using P0 with mass
  initial temperature = 300
  Calculate Volume From Enclosing Surface
End
```

You may also want to be able to set up a bulk node that just has a constant temperature on it (volume is still required, although not directly used here). A similar effect can also be obtained by just making the bulk element volume very large (or by increasing density or specific heat) and keeping the `bulk EQ` line.

```
Begin bulk fluid element bulk_volume_name
  material = a_bulk_volume
  initial temperature = 300
  bulk element volume = constant v = 0.25
End
```

Boundary Conditions

Bulk nodes can be linked with either `BC BULKNODE_FLUX` boundary conditions, or the older block-style boundary conditions. Boundary conditions linked to a bulk node generally behave the same way as their non-bulk-node versions, except that any required reference temperature now comes from the bulk node, and the flux lost from the boundary condition is added to the bulk node automatically.

For all equations boundary conditions may be applied by using a line command of the form:

```
BC BULKNODE_FLUX FOR energy ON surface_name = generalized_nat_conv BULK_
 ↪NODE=my_bulk_node
BC BULKNODE_FLUX FOR energy ON surface_name = generalized_rad BULK_
 ↪NODE=my_bulk_node
```

The older block-style boundary conditions for bulk nodes also work, although using the line commands is recommended. For the block-style boundary conditions, add the `Use Bulk Element` command to link it to a bulk node.

```
Begin Convective Flux Boundary Condition
  Use Bulk Element my_bulk_node
  .
End
```

In some cases it is desirable to couple equations being solved for different degrees of freedom on the volume and the bulk fluid element. For example coupling a porous media equation for the gas phase temperature on a volume block to an overall temperature on the bulk fluid element, or an equation solved for enthalpy to an equation solved for temperature. If the same degree of freedom is not present on the bulk node and the volume then the following mappings are checked for:

- Same degree of freedom (e.g. temperature, pressure, mass fraction) but without a material phase specification on the bulk node when one is present on the volume block.
- Temperature degrees of freedom on the volume will check for enthalpy in either the same material phase or no material phase on the bulk node (and vice versa for an enthalpy degree of freedom on the volume). Additional mappings can be supported where they make sense, please email sierra-help@sandia.gov if you have such a use case.

If either no valid mapping or multiple valid mappings are found the simulation will terminate during initialization.

In addition to boundary conditions coupling a bulk fluid element to the surface of a volumetric block it is possible to directly couple bulk fluid elements to one another via the bulk node coupling block. One common use case for this is *coupling between pressurization zones*.

Restarting With Bulk Fluid Element

The bulk fluid element capability is supported for *restarted simulations*. It is worth noting that while bulk fluid element is generally not part of the original mesh discretization the element is being created internal to the code and is also being added to the results file as well as the restart file if defined. In some cases one may wish to initialize a simulation from a previous restart file that does not contain the bulk fluid element. In this case provision must be made to ignore the fact that the initialization does not include the bulk fluid element so that the bulk fluid element can be created anew.

Input for restart jobs subsequent to the bulk fluid element being added to a model *should not* contain the BULK NODES IGNORE RESTART command line.

Advective Bars

For problems in which convection heat transfer occurs at the surface the analyst must supply some judgment about the fluid flow that gives rise to convection. While this information is usually obtained by performing a coupled conjugate heat transfer analysis including details of the fluid flow, in many cases the cost of such an analysis may be prohibitive so the analyst must resort to alternative lower order models. One such model, the advective bar, can be applied to cases in which the convective heat transfer is driven by a flow internal to the body of interest as in Fig. 4.17. The bar model includes a simplified description of forced flow and flow section along the length of a one-dimensional discretization, where the advected flow is directly coupled with a corresponding energy equation. The bar temperatures define the far-field temperature distribution for convective heat transfer to the two or three-dimensional thermal solid model. The advantage gained by using the simplified velocity and temperature representation is that the velocity can be used to define both laminar and turbulent convective heat transfer appropriate to the specific problem using empirical heat transfer coefficients (see *Correlation Heat Transfer Coefficient*).

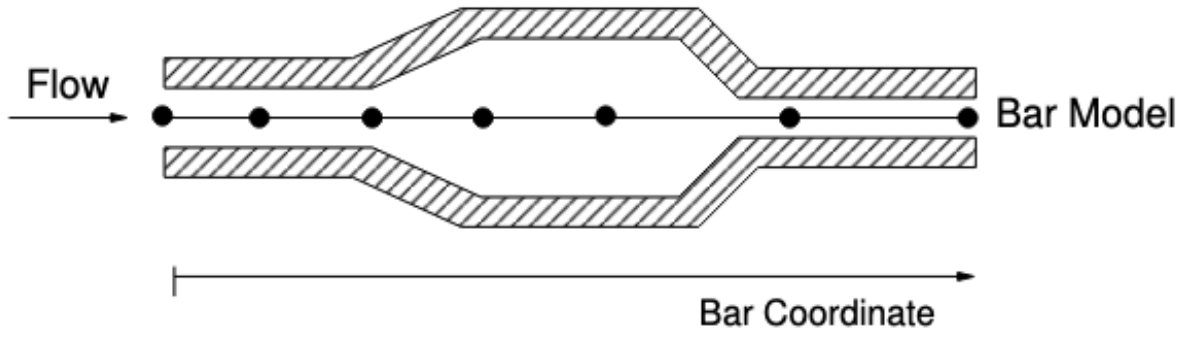


Fig. 4.17: Schematic of an advective bar

Using an Advective Bar as a Convective Flux BC

Use of the advective bar model requires that it be referenced by name in a Convective Flux Boundary Condition along with the corresponding named Heat Transfer Coefficient Correlation. Below shows a sample of an input deck that uses an advective bar `LIQUIDBAR` with a corresponding heat transfer correlation coefficient block `CHANNEL` to define a pair of convective flux boundary conditions, `CONV1` and `CONV2`.

```
BEGIN ARIA REGION MY_REGION
  .
  BEGIN ADVECTIVE BAR NETWORK LIQUIDBAR
    .
  END ADVECTIVE BAR NETWORK LIQUIDBAR
  .
  BEGIN HEAT TRANSFER CORRELATION COEFFICIENT CHANNEL
    .
  END HEAT TRANSFER CORRELATION COEFFICIENT CHANNEL
  .
  BEGIN CONVECTIVE FLUX BOUNDARY CONDITION CONV1
    .
  END CONVECTIVE FLUX BOUNDARY CONDITION CONV1

  BEGIN CONVECTIVE FLUX BOUNDARY CONDITION CONV2
    .
  END CONVECTIVE FLUX BOUNDARY CONDITION CONV2
.
END
```

Advective Bar Equation discusses the formulation of the advective bar equations. Fig. 4.18 shows a schematic of an advective bar block, labeled `Block_3` (black), set up between two volumetric blocks, `Block_1` (red) and `Block_2` (green). The surfaces that are coupled to the advective bar block are `Surface_1` (boxed in the red dashed line) and `Surface_2` (boxed in the green dashed line), which belong to `Block_1` and `Block_2` respectively. In addition, a nodelist is `Nodelist_1` is defined and contains only the left-most node in the advective bar block.

Defining an advective bar requires various line commands in the aria material block as well as individual region-level commands. An example of declaring a typical advective bar block for the Schematic shown in Fig. 4.18 is given in the following. We first declare functions that will represent the wetted perimeter and bar area at the Sierra scope. We will also define an Aria material that will be assigned to the bar block:

```
BEGIN DEFINITION FOR FUNCTION AREA_FUNC
  TYPE = PIECEWISE LINEAR
  BEGIN VALUES
     0.0 1.0
```

(continues on next page)

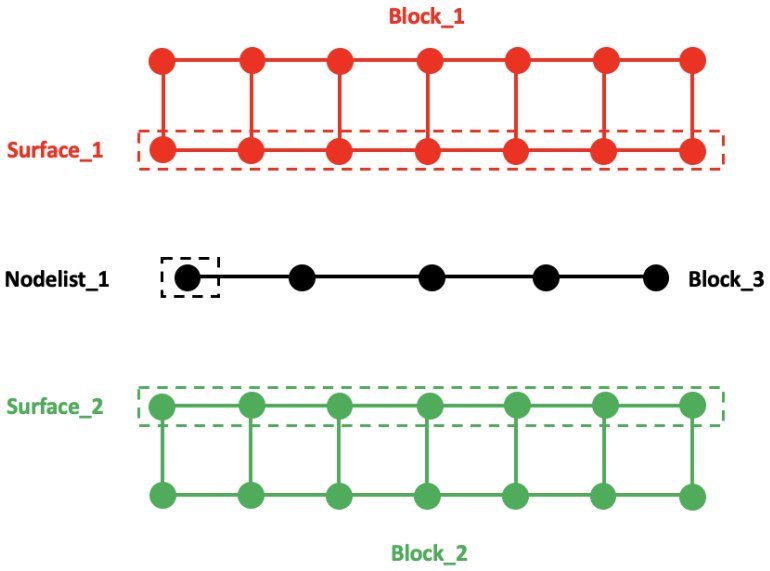


Fig. 4.18: Schematic of an advective bar block set up between two mesh blocks

(continued from previous page)

```
    10.0 1.0
  END VALUES
END DEFINITION FOR FUNCTION AREA_FUNC

BEGIN DEFINITION FOR FUNCTION WETTED_PERIM_FUNC
  TYPE = PIECEWISE LINEAR
  BEGIN VALUES
     0.0  0.5
    10.0  0.5
  END VALUES
END DEFINITION FOR FUNCTION WETTED_PERIM_FUNC

BEGIN ARIA MATERIAL WATER
  .
  ADVECTION VELOCITY = BAR
  BAR AREA = USER_FUNCTION NAME = AREA_FUNC X = BAR_COORDINATE
  DENSITY = constant rho = 1000.0
  .
END ARIA MATERIAL WATER

BEGIN FINITE ELEMENT MODEL FEM
  .
```

(continues on next page)

(continued from previous page)

```
  USE MATERIAL WATER FOR BLOCK_3
  .
END
```

The advection velocity for the bar material WATER must be defined as ADVECTION VELOCITY = BAR. This advection velocity will be used for the bar energy (and possibly momentum) equation. The advective bar command block along with the bar energy equation are defined as follows:

```
BEGIN ARIA REGION MY_REGION
  BEGIN EQUATION SYSTEM ENERGY
    .
    EQ ENERGY FOR TEMPERATURE ON BLOCK_3 USING Q1 WITH DIFF LUMPED_MASS␣
↪ADV SUPG
    ADVECTION VELOCITY FOR ENERGY = ADVECTION_VELOCITY
    .
    BEGIN ADVECTIVE BAR NETWORK LIQUIDBAR
      .
      ADD VOLUME BLOCK_3
      MASS FLOW RATE AT NODELIST_1 = 100.0 # KG/SEC^2
      FLUID DENSITY = 1000.0
      FLOW CROSS SECTION AREA FUNCTION = AREA_FUNC
      WETTED PERIMETER FUNCTION = WETTED_PERIM_FUNC
      .
    END ADVECTIVE BAR NETWORK LIQUIDBAR
  .
  END
END ARIA REGION MY_REGION
```

In the above, the advective bar area and density is defined under both the Aria material WATER and advective bar network LIQUIDBAR. The reason why this is done is discussed in *Current Discrepancies with Advective Bars*. The nodelist block block_3 is added to the simulation as an advective bar. For the above command block, the fluid density and fluid cross section area are declared as constants. The leftmost node on the advective bar (nodelist_1) is assigned a mass flow rate. A HEAT TRANSFER CORRELATION COEFFICIENT block is declared under the same equation system (ENERGY for this example), labeled CHANNEL:

```
BEGIN HEAT TRANSFER CORRELATION COEFFICIENT CHANNEL
  LAMINAR CORRELATION = TYPE 1
  TURBULENT CORRELATION = TYPE 74
  TRANSITION REYNOLDS NUMBER = 3000.
  COMPUTE REYNOLDS NUMBER
  COMPUTE PRANDTL NUMBER
  COMPUTE CHARACTERISTIC LENGTH MODEL = HYDRAULIC DIAMETER
```

(continues on next page)

(continued from previous page)

```
  COMPUTE FRICTION FACTOR MODEL = SMOOTH TUBE
  COMPUTE ANNULUS DIAMETER RATIO
END
```

The HEAT TRANSFER CORRELATION COEFFICIENT defines a Nusselt Number (Nu) which is used to compute the heat transfer correlation coefficient (Eq. (3.80)). The above declaration uses a fully developed laminar flow with an analytical Nu for the laminar region and a Gnielinski-type correlation for annular flow with a friction factor in the turbulent regime. The details of declaring the above command block can be found in *Correlation Heat Transfer Coefficient*. Convective boundary conditions, labeled CONV1 and CONV2 are defined for surfaces surface_1 and surface_2 in Fig. 4.18, respectively:

```
BEGIN CONVECTIVE FLUX BOUNDARY CONDITION CONV1
  ADD SURFACE Surface_1
  USE CORRELATION CONVECTION MODEL CHANNEL
  USE ADVECTIVE BAR LIQUIDBAR
END CONVECTIVE FLUX BOUNDARY CONDITION CONV1

BEGIN CONVECTIVE FLUX BOUNDARY CONDITION CONV2
  ADD SURFACE Surface_2
  USE CORRELATION CONVECTION MODEL CHANNEL
  USE ADVECTIVE BAR LIQUIDBAR
END CONVECTIVE FLUX BOUNDARY CONDITION CONV2
```

Alternatively, a single line command can be used in place of the above command block:

```
BC BAR_FLUX FOR ENERGY ON Surface_1 = GENERALIZED_NAT_CONV BAR=LIQUIDBAR␣
 ↪HCORR = CHANNEL
BC BAR_FLUX FOR ENERGY ON Surface_2 = GENERALIZED_NAT_CONV BAR=LIQUIDBAR␣
 ↪HCORR = CHANNEL
```

It should be noted that the above advective bar commands can also be coupled to other energy EQ types, such as POROUS_ENTHALPY.

A bar model coordinate system facilitates user specification of the flow section properties. Here the bar coordinate system can be aligned with one of a Cartesian coordinate X,Y,Z directions. Optional use of an S coordinate system tied directly to the bar discretization enables modeling of flow along a circuitous path. Entrance length effects are supported for a limited number of empirical correlations. By default, influence of the entrance length effect is aligned with the bar model coordinates. Cases in which these effects are significant only over a portion of the flow can be modeled using an offset of the bar model coordinate (a schematic of this is shown in Fig. 4.19).

Advective Bar contains details about available commands for declaring advective bars.

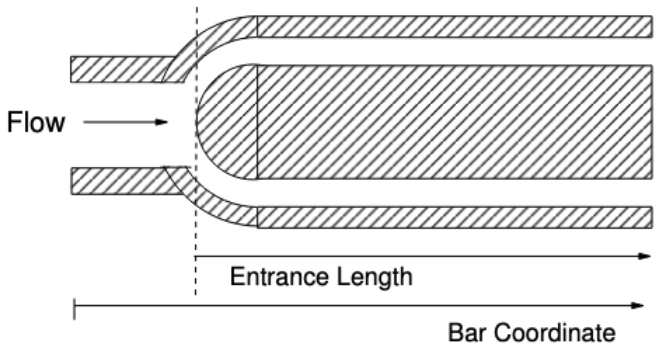


Fig. 4.19: Schematic of entrance length w.r.t bar coordinates

Beta Capability
BAR MOMENTUM EQUATION is a beta feature and requires use of the –beta flag. This capability is not thoroughly tested and should be used with caution. Refer to *Current Discrepancies with Advective Bars* for proper usage.

Pressure Solve for Advective Bars

The pressure can be solved over the advective bar when requested. This is done by defining an equation system for the bar block:

```
BEGIN EQUATION SYSTEM Pressure
  .
  EQ BAR_MOMENTUM FOR PRESSURE ON BLOCK_3 USING Q1 WITH ADV SRC
  SOURCE FOR BAR_MOMENTUM ON BLOCK_3 = MAJOR_LOSS
  ADVECTION VELOCITY FOR BAR_MOMENTUM = ADVECTION_VELOCITY
  BC DIRICHLET FOR PRESSURE ON NODELIST_1 = CONSTANT VALUE = 0.0
  .
END
```

The individual terms ADV and SRC are defined in *Advective Bar Pressure*. A MAJOR_LOSS source term is defined, which is the Darcy-Weisbach equation for modeling head loss due to friction (Eq. (3.86)). Because of this, we need to define a friction factor model in the Aria material block as well:

```
BEGIN ARIA MATERIAL WATER
  .
  FRICTION FACTOR = HALAAND ROUGHNESS = 3.0E-5
  .
END ARIA MATERIAL WATER
```

A list of Aria material friction factors can be found in *Friction Factor*. It should be noted that these friction factors are used separately from the HEAT TRANSFER CORRELATION COEFFICIENT computed friction factors; their discrepancies are discussed in *Current Discrepancies with Advective Bars*.

Current Discrepancies with Advective Bars

The following presents some discrepancies with defining advective bars that users should be aware of; these discrepancies will be resolved in time. Currently the advective bar command block is defined as follows within the Aria region scope:

```
BEGIN ADVECTIVE BAR NETWORK LIQUIDBAR
  .
  ADD VOLUME BLOCK_3
  MASS FLOW RATE AT NODELIST_1 = 100.0 # KG/SEC^2
  FLUID DENSITY = 1000.0
  FLOW CROSS SECTIONAL AREA FUNCTION = AREA_FUNC
  WETTED PERIMETER FUNCTION = WETTED_PERIM_FUNC
  .
END ADVECTIVE BAR NETWORK LIQUIDBAR
```

The fluid density and the bar area must be repeated in the Aria materials block assigned to the bar:

```
BEGIN ARIA MATERIAL WATER
  .
  BAR AREA = USER_FUNCTION NAME = AREA_FUNC X = BAR_COORDINATE
  DENSITY = constant rho = 1000.0
  .
END ARIA MATERIAL WATER

BEGIN FINITE ELEMENT MODEL FEM
  .
  USE MATERIAL WATER FOR BLOCK_3
  .
END
```

Each of these two duplicate parameters (bar area and density) are used for separate purposes; the FLUID DENSITY and FLOW CROSS SECTIONAL AREA FUNCTION under the advective bar command block are used for computing the nodal bar velocity vector *output* (default name is BAR_VELOCITY) or imposing a mass flow rate over the bar from a velocity boundary condition. The Aria material (WATER for our example) density and bar area is used to compute the velocity at the bar element integration points (Eq. (3.79)), which is used as the advecting velocity (default name is ADVECTION_VELOCITY) in the bar energy (and momentum) equation. The nodal

BAR_VELOCITY is currently automatically provided, whereas the integration point ADVECTION_VELOCITY must be postprocessed for visualization (see *Postprocessing*). For cases where these properties differ (and some other corner cases with sharp angular turns in the bar block) the BAR_VELOCITY and the postprocessed ADVECTION_VELOCITY may differ. For visualization, it is recommended that the user request the magnitude of the bar velocity vector, BAR_VELOCITY_MAG, which can be postprocessed as an expression.

Additionally, the FLOW CROSS SECTIONAL AREA FUNCTION function under the advective bar command block can be used to compute the hydraulic diameter by using the WETTED PERIMETER FUNCTION for the HEAT TRANSFER CORRELATION COEFFICIENT command block.

For computing the pressure via the BAR_MOMENTUM equation, a friction factor must be defined at the Aria material level if the MAJOR LOSS source term accounting for viscous effects is used:

```
BEGIN ARIA MATERIAL WATER
  .
  FRICTION FACTOR = HALAAND ROUGHNESS = 3.0E-5
  .
END ARIA MATERIAL WATER
```

In addition to this, a friction factor model is defined within HEAT TRANSFER CORRELATION COEFFICIENT command blocks to compute a heat transfer coefficient:

```
BEGIN HEAT TRANSFER CORRELATION COEFFICIENT CHANNEL
  .
  COMPUTE FRICTION FACTOR MODEL = SMOOTH TUBE
  .
END
```

It should be noted that these friction factors are used for separate computations. In the current implementation of the BAR_MOMENTUM equation, only the HALAAND friction factor model can be defined with the MAJOR_LOSS source term; if any other friction factor model is defined along with a friction factor model under the HEAT TRANSFER CORRELATION COEFFICIENT command block, then a parsing error will occur with the message being related to multiple expression handles being defined for the friction factor.

Directed Energy

Aria contains a number of directed energy (laser) source and flux models. The BEGIN DIRECTED ENERGY block in the Aria Region provides a unified way of specifying these sources and fluxes for a given configuration.

A directed energy model can be implemented as a surface flux, a source that penetrates into the surface, or a combination of the two. The typical inputs for a directed energy model are:

- LASER POWER - The total power of the laser in Watts (for problems in SI MKS units). If

there are multiple sources/fluxes specified the total power is divided among them based on their individual efficiency values (which must sum to 1).

- `PATH FUNCTION` - A piecewise linear function for the time-varying position and orientation of the laser **source**.
- `ORIGIN AND NORMAL FUNCTION` - A piecewise linear function for the time-varying position and orientation of the laser **flux**.
- `BEGIN X SOURCE` - A sub-block defining a source model to use and its relevant parameters.
- `BEGIN X POWER WELD FLUX` - A sub-block defining a flux model to use and its relevant parameters.

The complete set of commands for the directed energy model can be found in the *directed energy command reference*. An example input block is shown below for a case using both a source term and flux term in the same block.

```
BEGIN DIRECTED ENERGY MODEL bc2
  laser_power = CONSTANT Value = 1500

  add volume block_1    # for the source
  path function = path1 # for the source

  add surface surface_1 # for the flux
  Origin and Normal function = path2 # for the flux

  BEGIN Sharp Power Weld Flux
    efficiency = 0.5
    beam diameter = 0.20
  END

  BEGIN hemisphere source
    efficiency = 0.5
    beam diameter = 0.20
  END
END
```

Laser Path

The movement of the laser source and flux are controlled by multicolumn user-defined functions, in the format shown below.

```
Begin Definition for Function my_path
  Type is multicolumn piecewise linear
```

(continues on next page)

(continued from previous page)

```
  Column Titles Time X Y Z nx ny nz
  Begin Values
    0.0   2.2e-2  5e-3  1.5e-2 0 1 0
    4.0   2.2e-2  1e-3 -1.5e-2 0 1 0
    8.0   2.2e-2  1e-3 -1.5e-2 0 1 0
  End
End
```

The laser sources/fluxes will follow the path defined by this function and the first and last rows of the table define the start and end times and positions. Outside those times, the laser is off and there is no source or flux.

Note: Both the `PATH FUNCTION` and `ORIGIN AND NORMAL FUNCTION` expect functions of this format. The `PATH FUNCTION` is used for **sources** while `ORIGIN AND NORMAL FUNCTION` is used for boundary **fluxes**.

Note: For laser **fluxes** the path normal vector should be the direction of the laser, and its orientation (towards or away from the surface) does not matter. The laser path origin should be the point where the laser originates from (so the laser vector is defined by the point and normal).

Note: For laser **sources**, the normal is optional and is used to define the source depth direction, and it should be pointed *away* from the surface (so the source depth is in the opposite direction as the normal). The laser path point is the center of the source (e.g. the center of the sphere for a spherical source).

Source Models

The various types of laser sources available are described in the following sections. For a complete list of the commands required for each model, refer to the *directed energy command summary*.

Hemispherical

The hemisphere model defines a hemispherical volume to heat the material block. If a Gaussian distribution is selected, the model is defined as:

$$q_{hemisphere} = \frac{6\sqrt{3}\eta P}{r_{beam}^3 \pi \sqrt{\pi}} \exp\left(-3\left(\frac{r^2}{r_{beam}^2}\right)\right)$$

where r_{beam} is the beam radius, r is the minimum distance from the evaluation point to the laser focal spot, η is the efficiency and P is the laser power. If the uniform distribution is selected, then the model is defined as:

$$q_{hemisphere} = \frac{3\eta P}{2\pi r_{beam}^3}$$

The figure below shows a schematic of the hemispherical heating source.

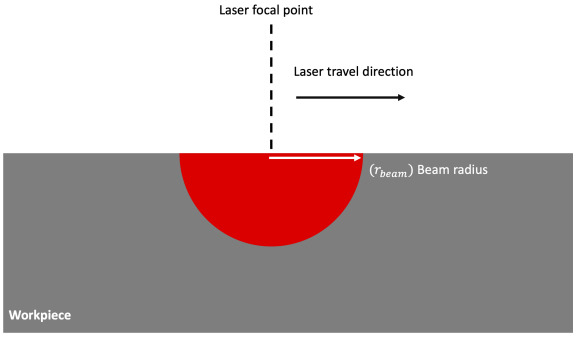


Fig. 4.20: Hemispherical source geometry.

Spherical

The sphere model defines a spherical volume to heat the material block. If a Gaussian distribution is selected, the model is defined as:

$$q_{sphere} = \frac{3\sqrt{3}\eta P}{r_{beam}^3 \pi \sqrt{\pi}} \exp\left(-3\left(\frac{r^2}{r_{beam}^2}\right)\right)$$

where r_{beam} is the beam radius, r is the minimum distance from the evaluation point to the laser focal spot, η is the efficiency and P is the laser power. If the uniform distribution is selected, then the model is defined as:

$$q_{sphere} = \frac{3\eta P}{4\pi r_{beam}^3}$$

The figure below shows a schematic of the spherical heating source.

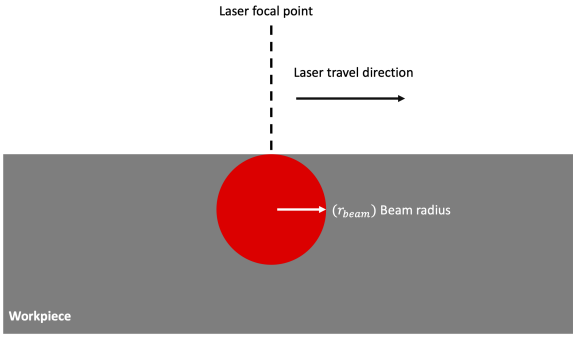


Fig. 4.21: Spherical source geometry.

Beer-Lambert

The Beer-Lambert model assumes an exponential decrease of the laser absorptivity with penetration depth. It is defined as:

$$q_{BL} = \frac{\eta r_{eff} P}{\pi r_{beam}^2 \beta^2 (1 - \exp(-r_{eff}))} \exp\left(-r_{eff}\left(\frac{r_p^2}{r_{beam}^2}\right)\right) I_{att}$$

where r_{eff} the effective beam radius, r_{beam} is the beam radius P is the laser power, β is a spatial factor, r_p is the planar distance from the evaluation point to the laser focal point and I_{att} is an attenuation factor that varies exponentially with depth. The spatial factor β is an optional parameter defined by SPATIAL INFLUENCE FACTOR, and is given a default value of 1. The attenuation factor I_{att} is defined as:

$$I_{att} = \frac{\alpha \exp(-(z - z_c)\alpha)}{1 - \exp(-\alpha z_{max})}$$

where α is an absorption coefficient $(1/L)$ of range roughly 1 to 1e6 defined as a material property for the volume of interest. The value z_c is the z position of the laser focal point and z_{max} is the maximum penetration depth, which is currently defined as $z_{max} = 6r_{beam}$. Surface absorption of energy can be accounted for by defining a modified attenuation factor ηI_{att} where η is the efficiency parameter.

Parallelpiped

The parallelpiped model is defined as:

$$q_{parallelpiped} = \frac{\eta P}{4abc}$$

where a is the travel semi axis, b is the width semi axis, c is the penetration depth, η is the efficiency and P is the laser power. The parallelpiped assumes a uniform distribution. The figure below shows a schematic of the parallelpiped heating source.

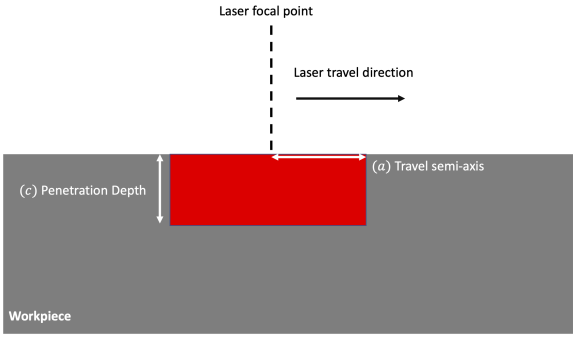


Fig. 4.22: Parallelpiped source geometry.

Ellipsoid

The ellipsoid model is defined as:

$$q_{ellipsoid} = \frac{6\sqrt{3}\eta P}{abc\pi\sqrt{\pi}} \exp\left(-3\left(\frac{(x-x_c)^2}{a^2} + \frac{(y-y_c)^2}{b^2} + \frac{(z-z_c)^2}{c^2}\right)\right)$$

where a is the travel semi-axis, b is the width semi-axis, c is the depth semi-axis, η is the efficiency, P is the laser power and x_c, y_c, and z_c are the x, y, and z coordinate of the laser focal point, respectively. The figure below shows a schematic of the ellipsoid heating source, where the width semi-axis is pointing out of the page.

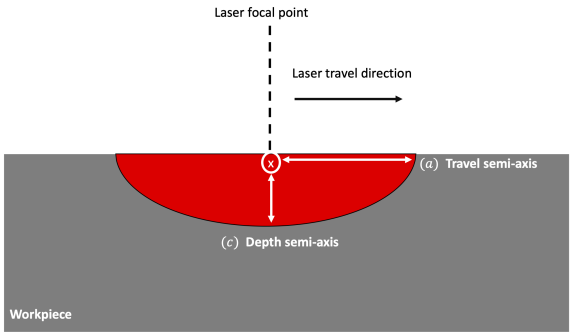


Fig. 4.23: Ellipsoid source geometry.

Double Ellipsoid

The double ellipsoid model defines two ellipsoid energy sources, $q_{ellipsoid,fore}$ and $q_{ellipsoid,aft}$ combined at the laser focal point. The fore ellipsoidal energy source is defined as:

$$q_{ellipsoid,fore} = \frac{6\sqrt{3}\eta P f_{fore}}{a_{fore} bc\pi\sqrt{\pi}} \exp\left(-3\left(\frac{(x-x_c)^2}{a_{fore}^2} + \frac{(y-y_c)^2}{b^2} + \frac{(z-z_c)^2}{c^2}\right)\right)$$

where a_{fore} is the fore travel semi-axis, b is the width semi-axis, c is the depth semi-axis, f_{fore} is the fore power fraction, η is the efficiency P is the laser power, and x_c, y_c and z_c are the x, y and z coordinate of the laser focal point, respectively. The aft ellipsoidal energy source is defined as:

$$q_{ellipsoid,aft} = \frac{6\sqrt{3}\eta P f_{aft}}{a_{aft} bc\pi\sqrt{\pi}} \exp\left(-3\left(\frac{(x-x_c)^2}{a_{aft}^2} + \frac{(y-y_c)^2}{b^2} + \frac{(z-z_c)^2}{c^2}\right)\right)$$

where a_{aft} is the fore travel semi-axis and f_{aft} is the aft power fraction. The figure below shows a schematic of the double ellipsoid energy source model, where the width semi-axis is pointing out of the page.

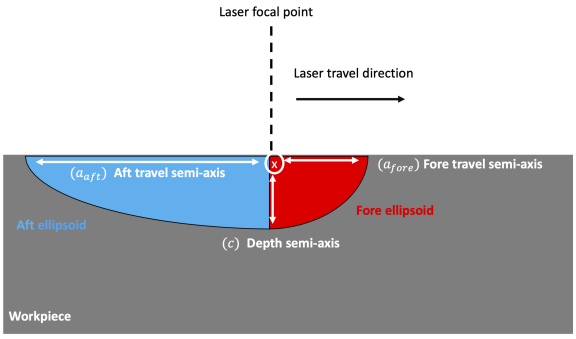


Fig. 4.24: Double ellipsoid source geometry.

Conical

The conical heat source model is defined as:

$$q_{conical} = \frac{9\exp(3)\eta}{\pi d(\exp(3)-1)(r_s^2 + r_s r_d + r_d^2)} \exp\left(\frac{-r_p^2}{r_0^2}\right)$$

where η is the efficiency, d is the penetration depth, r_s is the beam radius at the laser focal point, r_d is the beam radius at penetration depth d and r_p is the planar distance from the evaluation point to the laser focal point. The term r_0 is the beam radius at a particular depth z, and is defined as:

$$r_0 = r_s - (r_s - r_d)\left(\frac{z - z_c}{d}\right)$$

where z_c is the z position of the laser focal point. The figure below shows a schematic of the conical energy source model.

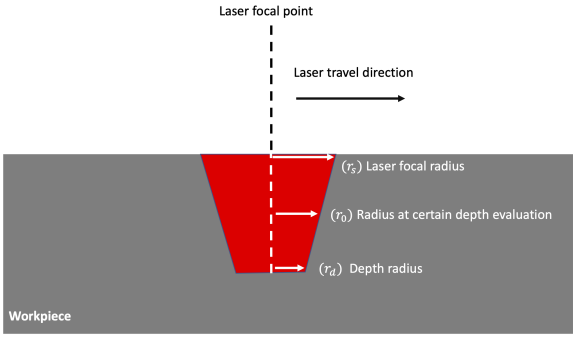


Fig. 4.25: Conical source geometry.

Flux Models

The various types of laser flux models available are described in the following sections. For a complete list of the commands required for each model, refer to the *directed energy command summary*.

Sharp

The coordinates of the center of the beam are provided by the X, Y, and Z components of the provided origin_and_normal_function. The direction of the beam is provided by the NX, NY, and NZ components of the origin_and_normal_function. The applied flux is given by:

$$\frac{(-\boldsymbol{n}) \cdot \boldsymbol{d}}{|\boldsymbol{d}|} \frac{\alpha e P}{\pi R^2}$$

P is the laser power defined for the material, R is the beam radius, α is the laser absorptivity defined for the material, e is the EFFICIENCY, and the term $-\boldsymbol{n} \cdot \boldsymbol{d}$ accounts for the fact that the surface may not be orthogonal to the laser. Points with values of r_p > CLIP_RADIUS will receive zero flux. r_p is the shortest distance between a point on the surface and the line defined by the beam origin and normal. If ACTIVATION is set to true, points whose temperature exceed ACTIVATION_TEMPERATURE will be flagged as activated for use with the ACTIVATION USER FUNCTION thermal conductivity model. If INTEGRATED is true, this flux will be integrated between the previous time step and the current one and the average flux applied.

Gaussian

The coordinates of the center of the beam are provided by the X, Y, and Z components of the provided origin_and_normal_function. The direction of the beam is provided by the NX, NY, and NZ components of the origin_and_normal_function. The applied flux is given by:

$$\frac{(-\boldsymbol{n})\cdot\boldsymbol{d}}{|\boldsymbol{d}|}\frac{\alpha e R_{eff} P}{\pi R^2(1-e^{-R_{eff}})}e^{-R_{eff}\frac{r_p^2}{R^2}}$$

P is the laser power defined for the material, R is the beam radius, R_{eff} is the effective beam radius, α is the laser absorptivity defined for the material, e is the EFFICIENCY, and r_p is the shortest distance between a point on the surface and the line defined by the beam origin and normal. The term $-\boldsymbol{n}\cdot\boldsymbol{d}$ accounts for the fact that the surface may not be orthogonal to the laser. Points with values of r_p > CLIP_RADIUS will receive zero flux. If ACTIVATION is set to true, points whose temperature exceed ACTIVATION_TEMPERATURE will be flagged as activated for use with the ACTIVATION USER FUNCTION thermal conductivity model. If INTEGRATED is true, this flux will be integrated between the previous time step and the current one and the average flux applied.

Feature Toggling

For transient analysis Aria provides a mechanism for enabling and disabling certain aspects of a model like boundary conditions, contact conditions, source contributions and element block physics during selected time periods. This option is useful when modeling a time sequence of discontinuous events and the capability will be subsequently referred to as “toggling”. Toggling as applied to simple electric switches means that one either opens or closes the switch (i.e. the switch is on or off). Similarly, from an analysis perspective one simply defines a specific model feature to be either active or inactive. Toggling **does not** imply the ability to swap in another physics model or to apply different model parameters.

Generally speaking the use of toggling is justified when a change in model physics will result in a change to the structure of the system matrix or to the structure of linear system contributions. For example, change of a flux boundary condition to a Dirichlet boundary condition requires a change to the structure of the matrix contribution since unknown values are now specified and need not be computed. Similarly, deactivation of a element block will remove degrees-of-freedom (rows and columns) from the Jacobian matrix thus reducing the number of system unknowns. Note that a change of flux boundary condition input parameters (e.g. the convective coefficient) will modify the system matrix coefficients but not affect the structure of the associated linear system matrix contributions so feature toggling is not justified in this case. Instead parameter changes in time should be accounted for through the parameter model definition.

Setting up toggling requires two specifications in the input deck. First, the active/inactive state history of a feature is defined using a TOGGLE BLOCK command block at the SIERRA scope. This command block specifies whether associated features should be active or inactive during each transient block of the simulation. Second, the features that are being toggled must be associated with the corresponding toggle block.

Contact, source terms, boundary conditions, and initial conditions defined via block commands are associated with a toggle block via the USE TOGGLE BLOCK command line. Source terms, boundary conditions, and initial conditions defined via single line commands are associated with a toggle block via an argument "toggle = toggle_block_name" at the end of the line command. Toggling of an entire element block is established via a standalone USE TOGGLE BLOCK command line within the Region scope. For toggling of the element block associated with a *bulk fluid element* we note that the corresponding element block name is BLOCK_FOR_bulk_element_name.

Toggling is supported for flux boundary conditions, contact conditions and source terms by simply zeroing out their contributions. Thus the number of DOF in the linear system will not change. Toggling of Dirichlet boundary conditions and element blocks is currently supported by removing the element DOF from the linear system.

Oftentimes one may wish to toggle off a surface boundary condition and toggle on a variant of the same boundary condition. As an example, one might replace a convective flux boundary condition with constant reference temperature with one having a reference temperature defined by a tabular function. Since the original boundary condition still exists (but it's toggled off) the reference temperature definition will conflict with the new boundary condition. Cases such as this will require that the second boundary condition be applied to a different surface (differently named) that overlies the surface used in the original boundary condition in order to resolve this conflict.

Note: Toggling is not supported for material properties.

Usage of toggling is demonstrated in the outline below where we note that the time period within the TOGGLE BLOCK corresponds to one of the time periods defined within the *Solution Control* command block. A full command reference is provided in the *command summary*.

```
Begin Sierra myJob
  .
  Begin Toggle Block first_toggle
    period = a_time_period_2
    state = active
  End
  .
  Begin Toggle Block second_toggle
    period = a_time_period_3
    state = inactive
  End
  Begin Toggle Block third_toggle
    period = a_time_period_3
    state = active
```

(continues on next page)

(continued from previous page)

```
End
.
Begin Procedure My_Aria_Procedure
  .
  Begin Solution Control Description
    Use system main
    Begin system main
      Simulation start time = 0.0
      Simulation termination time =  3600.0
      Begin transient a_time_period_1
        advance My_Region
      End
      Begin transient a_time_period_2
        advance My_Region
      End
      Begin transient a_time_period_3
        advance My_Region
      End
    End

    Begin parameters for transient a_time_period_2
      start time = 0.0
      termination time = 2.0
      Begin parameters for Aria region My_Region
      .
      End parameters for Aria region My_Region
    End parameters for transient a_time_period_2

    Begin parameters for transient a_time_period_2
      start time = 2.0
      termination time = 5.0
      Begin parameters for Aria region My_Region
      .
      End parameters for Aria region My_Region
    End parameters for transient a_time_period_2

  End
  .
  Begin Aria Region My_Region
    .
    Use toggle block first_toggle for block_8
    .
```

(continues on next page)

(continued from previous page)

```
      Begin Volume Heating h1
        .
        use toggle block first_toggle
      End
      Source for energy on block_1 = constant value=1 toggle=second_toggle
      Begin Heat Flux Boundary Condition bc1
        .
        Use toggle block second_toggle
      End
      BC Dirichlet for temperature on surface_3 = constant value = 400␣
↪toggle=third_toggle
      .
    End
    .
  End
  .
End Sierra myJob
```

Note that if one wishes to replace a boundary condition (the physics model) then one can do so by first adding a duplicate surface mesh entity and adding a new toggle block corresponding to usage of the new physics model.

It is important to note that within a TOGGLE BLOCK only one event (active or inactive) can be specified for the selected time period(s). Time periods other than the ones specified within the TOGGLE BLOCK command block will assume the opposite character (active/inactive). With reference to the previous outline this behavior is demonstrated by simple examples.

For the following toggle block

```
Begin Toggle Block my_toggle
 period = a_time_period_2
 state = active
End
```

a feature is active for a_time_period_2 and inactive for a_time_period_1 and a_time_period_3. Similarly the same behavior could be obtained using

```
Begin Toggle Block my_toggle
 period = a_time_period_1 a_time_period_3
 state = inactive
End
```

Level Set/CDFEM

Level set methods can be used to model multiphase flow where an evolving free surface must be accounted for. This section goes over how to set up a level set equation within Aria, as well as setting up a diffuse interface or sharp interface (CDFEM) approach.

Level Set Equation and Interface Definition

Within the Aria region, an input deck must define a level set equation system.

```
BEGIN EQUATION SYSTEM LS
  .
  [SOLVER SETTINGS]
  EQ LEVEL_SET FOR LEVEL_SET ON BLOCK_1 USING Q1 WITH MASS ADV SUPG
  .
END EQUATION SYSTEM
```

In the above input block, `[SOLVER SETTINGS]` includes any information about the nonlinear solution strategy, such as the maximum nonlinear iteration, nonlinear solution strategy etc. For the example above, the `MASS` and `ADV` terms are included. *Level Set* goes over the terms and formulation of the level set equation. SUPG stabilization is also added; information about SUPG stabilization can be found in *SUPG*. In addition to defining the level set equation system, a `LEVEL SET INTERFACE` must be defined in the Aria region scope.

```
BEGIN LEVEL SET INTERFACE LS
  .
  DISTANCE VARIABLE = SOLUTION->LEVEL_SET
  PERFORM INITIAL REDISTANCE = FALSE
  REDISTANCE METHOD = CLOSEST_POINT
  .
END LEVEL SET INTERFACE
```

The above shows some of the basic commands needed to define an interface, such as the distance variable and some commands associated with redistancing. Refer to *Level Set Interface* for more details on available commands for setting up a level set interface.

Redistancing the Level Set Field

As mentioned in *Level Set*, the advection of the level set field in general does not preserve the signed distance property. The integration of Eq. (3.91) can result in the formation of sharp and shallow gradients in the level set field Φ; these sharp and shallow gradients may result in numerical difficulties while solving Eq. (3.91) and inappropriate widening of the interface, respectively. To correct this, redistancing of the level set field is periodically done throughout the simulation. It should be noted that redistancing occurs in pseudo time, and not physical time. Aria utilizes Sierra/Krino to support two methods of redistancing the level set field: `CLOSEST_POINT` and `FAST_MARCHING` method. The default is the `CLOSEST_POINT` method.

The `FAST_MARCHING` was originally developed by Sethian [39] to solve boundary value problems of the Eikonal equation:

$$\begin{aligned} \|\nabla t_a(\boldsymbol{x})\| &= \frac{1}{F_{ext}} \quad \text{for} \quad \boldsymbol{x} \in V \\ t_a(\boldsymbol{x}) &= 0 \quad \text{for} \quad \boldsymbol{x} \in \Gamma \end{aligned} \tag{4.6}$$

where t_a is a time of arrival and F_{ext} is an extension speed of the interface; this type of equation describes the evolution of a closed interface Γ as a function of t_a with speed F_{ext} in the direction normal to Γ. Eq. (4.6) can be manipulated to look like the condition posed by Eq. (3.89):

$$\|\nabla \Phi\| = 1 \tag{4.7}$$

which can be solved via the `FAST_MARCHING` method. For a more detailed description of this method, users are referred to [39].

The `CLOSEST_POINT` method reconstructs a piecewise linear representation of the $\Phi = 0$ interface through elements that have differing signs of Φ. From this, facets representing the interface are constructed and each node i in the mesh finds a minimum distance D_i to this set.

$$\Phi_i^* = \text{sign}(\Phi_i^0) D_i \tag{4.8}$$

where Φ_i^0 is the level set function prior to redistancing and Φ_i^* is the newly redistanced level set. Given a sufficient density of facets, this procedure yields good results and is fast and robust.

Both `FAST_MARCHING` and `CLOSEST_POINT` redistancing methods can potentially perturb the $\Phi = 0$ interface resulting in a loss of mass. A volume constraint is enforced to ensure that the phase volume is conserved throughout the entire simulation. A global constant ε_Φ is solved from the following:

$$\int_{\mathrm{V}} H_A^0(\Phi^* + \varepsilon_\Phi) d\mathrm{V} = \int_{\mathrm{V}} H_A^0(\Phi^0) d\mathrm{V} \tag{4.9}$$

Where the sharp Heaviside function H_A^0 (indicated by a 0 superscript) is used to measure the volume of phase `A`. The final redistanced level set Φ_i is then computed by:

$$\Phi_i = \Phi_i^* + \varepsilon_\Phi \tag{4.10}$$

The redistancing options can be specified in the Solution Control Transient block. An example of performing redistancing after 7 physical time steps is shown below:

```
Begin Transient The_Time_Block_1
  .
  Event LS_CONSERVED_REDISTANCE when "(CURRENT_STEP - LAST_LS_CONSERVED_
↪REDISTANCE_STEP) >= 7"
  .
End
```

Where the `LS` in `LS_CONSERVED_REDISTANCE` uses the name of the `BEGIN LEVEL SET INTERFACE LS` block declared in *Level Set Equation and Interface Definition*. For the above example, `LS_CONSERVED_REDISTANCE` has been selected; this means that the initial volume measured in Eq. (4.9) is only measured once at the beginning of the simulation. For applications where mass of a particular phase is introduced into the simulation domain over time, the `LS_CONSTRAINED_REDISTANCE` event can be used, which measures the initial volume in Eq. (4.9) at the end of each physical time step before redistancing event occurs.

Diffuse Interface Approach

Within Aria, there are options for representing the interface as either diffuse or sharp. A diffuse interface, which is modeled with a smooth Heaviside function H_ε, varies the phase transition region (and therefore, phase properties) smoothly but sharply over a finite thickness. This thickness is typically related to the minimum element size in the mesh. The smooth Heaviside is evaluated as:

$$H_\varepsilon \equiv \begin{cases} 0 & \text{if } \Phi < -\varepsilon \\ \frac{1}{2}\left[1 + \frac{\Phi}{\varepsilon} + \frac{1}{\pi}\sin\left(\frac{\pi\Phi}{\varepsilon}\right)\right] & \text{if } |\Phi| \leq \varepsilon \\ 1 & \text{if } \Phi > \varepsilon \end{cases} \tag{4.11}$$

Here, the parameter ε controls the thickness of the interface, and is typically 2-3 times the minimum element size in the mesh. It can be seen that this smooth transition region between phases occurs in regions where $|\Phi| < \varepsilon$. An example of this plotted in 1D against a level set variable Φ where the interface is shown in Fig. 4.26. A interface thickness of $\varepsilon = 3\Delta x$ was chosen here, where Δx is the mesh size; Fig. 4.26 correspondingly shows that there are about 6 nodes across the interface thickness.

To set up a diffuse interface modeling approach within Aria, at least 3 material blocks must be declared; one for each phase `A` and `B` (for a single level set), and one multiphase material that combines the two `A` and `B` materials:

```
BEGIN ARIA MATERIAL POS_MAT
  .
  DENSITY = CONSTANT rho = 1
```

(continues on next page)

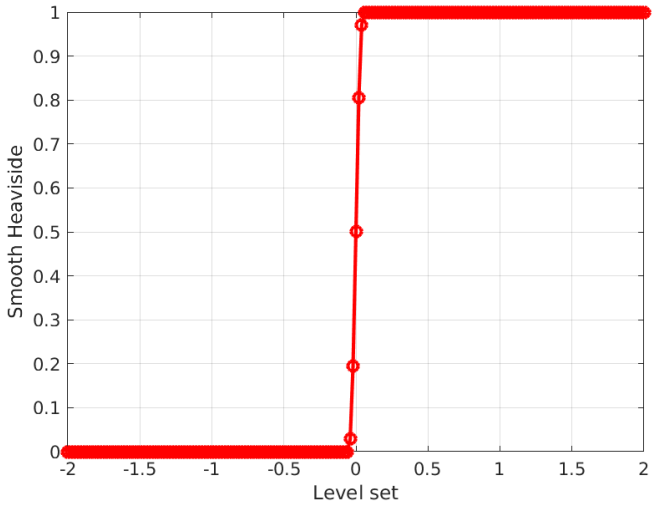


Fig. 4.26: Schematic of 1D smooth Heaviside as a function of Φ

(continued from previous page)

```
  VISCOSITY = CONSTANT MU = 1.0e-5
  .
END ARIA MATERIAL POS_MAT
.
BEGIN ARIA MATERIAL NEG_MAT
  .
  DENSITY = CONSTANT rho = 1000.0
  VISCOSITY = CONSTANT MU = 1.0e-3
  .
END ARIA MATERIAL NEG_MAT
.
BEGIN ARIA MATERIAL MULTIPHASE
  .
  LEVEL SET HEAVISIDE = INTERPOLATED
  LEVEL SET WIDTH = CONSTANT width = 0.05
  DENSITY = PHASE_AVERAGE
  VISCOSITY = PHASE_AVERAGE
  SURFACE TENSION = CONSTANT sigma = 0.3
  MOMENTUM STRESS = INCOMPRESSIBLE_NEWTONIAN
  .
END ARIA MATERIAL MULTIPHASE
```

In the above input block, materials `POS_MAT` and `NEG_MAT` are declared for regions where Φ are positive and negative, respectively. A third block, `MULTIPHASE`, is declared that defines the smooth Heaviside variable H_ε with the `LEVEL SET HEAVISIDE = INTERPOLATED` command and the interface thickness ε with the `LEVEL SET WIDTH` command. Available options for these

parameters can be found in *Material Properties*. It should be noted that in the above material definitions, a momentum stress model needs to be only defined for `MULTIPHASE`, and not `POS_MAT` or `NEG_MAT`. Lastly, material blocks are assigned within the `FINITE ELEMENT MODEL` scope. For a mesh that contains a `block_1` that will be separated into phases by the level set field, the following is needed to define a diffuse interface:

```
BEGIN FINITE ELEMENT MODEL MYFEM
  BEGIN PARAMETERS FOR BLOCK BLOCK_1
    .
    MATERIAL MULTIPHASE
    PHASE A = POS_MAT
    PHASE B = NEG_MAT
    .
  END
END
```

Sharp Interface Approach with CDFEM

Aria offers the option to model the Φ field as a sharp interface using Sierra/Krino. This is done through the conformal decomposition finite element method (CDFEM) developed by Noble et al. [40]. This is done by using the initial mesh as a background mesh, and cutting elements where the level set interfaces crosses into sub-elements to conform to the interface. The CDFEM approach to interface modeling will separate the original mesh into two distinct element blocks that represent each phase; the interface represented by the $\Phi = 0$ contour is also represented as a distinct surface within the mesh, which can be used for any interfacial boundary applications.

To set up a simulation with CDFEM, a set of material blocks similar to the diffuse interface approach in *Diffuse Interface Approach* is declared:

```
BEGIN ARIA MATERIAL POS_MAT
  .
  DENSITY = CONSTANT rho = 1
  VISCOSITY = CONSTANT MU = 1.0e-5
  MOMENTUM STRESS = INCOMPRESSIBLE_NEWTONIAN
  .
END ARIA MATERIAL POS_MAT
.
BEGIN ARIA MATERIAL NEG_MAT
  .
  DENSITY = CONSTANT rho = 1000.0
  VISCOSITY = CONSTANT MU = 1.0e-3
  MOMENTUM STRESS = INCOMPRESSIBLE_NEWTONIAN
  .
```

(continues on next page)

(continued from previous page)

```
END ARIA MATERIAL NEG_MAT
.
BEGIN ARIA MATERIAL POS_NEG_INTERFACE
  .
  SURFACE TENSION = CONSTANT SIGMA = 0.3
  .
END ARIA MATERIAL MULTIPHASE
```

In the above material blocks, the `MOMENTUM STRESS` model is defined for each individual phase; in addition the `MULTIPHASE` material block that was defined for the diffuse interface approach is removed. Instead, another material block `POS_NEG_INTERFACE` is defined, which will be the material block assigned to the surface block that represents the interface (which will be automatically generated by CDFEM).

Within the `FINITE ELEMENT MODEL` block, the following is defined for our example of a mesh with a single block `block_1` that contains the level set interface:

```
BEGIN FINITE ELEMENT MODEL MYFEM
  BEGIN PARAMETERS FOR PHASE POS
    where LS is positive
  END
  .
  BEGIN PARAMETERS FOR PHASE NEG
    where LS is negative
  END
  .
  BEGIN PARAMETERS FOR BLOCK BLOCK_1_POS
    Material POS_MAT
  END
  .
  BEGIN PARAMETERS FOR BLOCK BLOCK_1_NEG
    Material NEG_MAT
  END
  .
  BEGIN PARAMETERS FOR SURFACE SURFACE_BLOCK_1_POS_NEG
    Material POS_NEG_INTERFACE
  END
END
```

The `BEGIN PARAMETERS FOR PHASE` command defines the individual block phases which are separated by the level set interface `LS` (defined in *Level Set Equation and Interface Definition*). From this, Sierra/Krino will create decomposed blocks based on these phase names: `BLOCK_1_POS` and `BLOCK_1_NEG`. These decomposed blocks must have materials assigned to them. Lastly, a surface that represents the $\Phi = 0$ contour is created, which is named by combining

the block name and their phases; in our example, this is SURFACE_BLOCK_1_POS_NEG. This surface requires a material block defining interfacial properties to be defined on it as well (POS_NEG_INTERFACE in our example). An example of this is shown in Fig. 4.27, where the original BLOCK_1 mesh is shown on the left (and is kept as a background mesh by CDFEM); the decomposed parts BLOCK_1_NEG and BLOCK_1_POS are shown on the right, which are separated by a the surface SURFACE_BLOCK_1_POS_NEG.

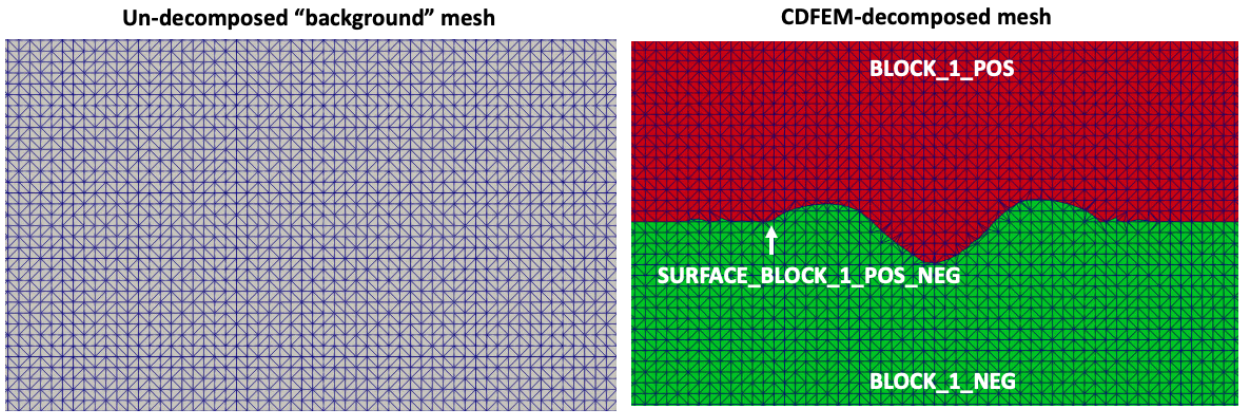


Fig. 4.27: Schematic of an original BLOCK_1 mesh decomposed into separate parts by CDFEM

In addition to defining a level set equation system on the original block (BLOCK_1), a similar set of equations must be assigned on the CDFEM-decomposed blocks as well:

```
EQ LEVEL_SET FOR LEVEL_SET ON BLOCK_1_NEG USING Q1 WITH MASS ADV SUPG
EQ LEVEL_SET FOR LEVEL_SET ON BLOCK_1_POS USING Q1 WITH MASS ADV SUPG
```

Surfaces that existed on the original mesh will also be decomposed in a similar manner. For our example, if there existed an original SURFACE_1, that surface will be decomposed into SURFACE_1_POS and SURFACE_1_NEG, which will correspond to the positive and negative phases, respectively.

A CDFEM OPTIONS block can be declared at the Aria region scope. An example of declaring a block like this is shown below:

```
BEGIN CDFEM OPTIONS
  .
  CDFEM EDGE TOLERANCE = 1.0e-2
  .
END
```

For a detailed list of CDFEM options and their purposes, refer to *CDFEM Options*.

Diffuse Level Set Approach to Modeling the Capillary Force

Aria offers various formulation for modeling the capillary (surface tension) force. The choice of the various capillary force implementations depend on the level set method used (diffuse or CDFEM); a result of this is that the momentum equation setup is also tailored to the selected level set approach. The following goes over setting up some of the basic implementations of the capillary forces; users should refer to the *Command References* for all available options.

To set up the momentum equation system, the following can be done (using the example from *Diffuse Interface Approach*):

```
EQ CONTINUITY_A FOR PRESSURE ON BLOCK_1 USING Q1 WITH DIV
EQ CONTINUITY_B FOR PRESSURE ON BLOCK_1 USING Q1 WITH DIV
EQ MOMENTUM_A FOR VELOCITY ON BLOCK_1 USING Q1 WITH DIFF ADV SRC SUPG
EQ MOMENTUM_B FOR VELOCITY ON BLOCK_1 USING Q1 WITH DIFF ADV SRC SUPG
```

The `CONTINUITY` and `MOMENTUM` equations are set up for both phases `A` and `B`. As mentioned in *Diffuse Interface Approach*, any discontinuities or jumps in the pressure or velocity field across phases `A` or `B` are smeared over the finite interface thickness.

In diffuse interface approaches, the surface tension force is represented as a continuous volumetric force, $\boldsymbol{F_\sigma}$; this is added as a source term to the momentum Eq. (3.23).

```
SOURCE FOR momentum ls A ON block_1 = LS_CAPILLARY
SOURCE FOR momentum ls B ON block_1 = LS_CAPILLARY
```

The `LS_CAPILLARY` source model is computed as (after taking a G/FEM residual form):

$$-\int_{\mathrm{V}} \boldsymbol{F_\sigma} \cdot \boldsymbol{e}_k \phi^i d\mathrm{V} \tag{4.12}$$

With $\boldsymbol{F_\sigma}$ taking the form:

$$\boldsymbol{F_\sigma} = \sigma \kappa \boldsymbol{n} \frac{\partial H_\varepsilon}{\partial \Phi} \tag{4.13}$$

where σ is a surface tension coefficient (which can be defined as a generic aria material property, refer to *Material Properties* for more details) and $\frac{\partial H_\varepsilon}{\partial \Phi}$ is:

$$\frac{\partial H_\varepsilon}{\partial \Phi} \equiv \begin{cases} 0 & \text{if } |\Phi| > \varepsilon \\ \frac{1}{2\varepsilon}\left[1 + \cos\left(\frac{\pi\Phi}{\varepsilon}\right)\right] & \text{if } |\Phi| \leq \varepsilon \end{cases} \tag{4.14}$$

A similar model that can be implemented for diffuse interface is the `LS_HEAVISIDE_CAPILLARY` model, which converts the product of $\boldsymbol{n}\frac{\partial H_\varepsilon}{\partial \Phi}$ in Eq. (4.13) into a Heaviside spatial gradient, ∇H_ε:

$$\boldsymbol{F_\sigma} = \sigma \kappa \nabla H_\varepsilon \tag{4.15}$$

CDFEM Approach to Modeling the Capillary Force

To set up the momentum equation system for a CDFEM approach requires a different format (using the example from *Sharp Interface Approach with CDFEM*, refer to Fig. 4.27 for a visual schematic):

```
EQ MOMENTUM FOR VELOCITY ON BLOCK_1_NEG USING Q1 WITH MASS ADV DIFF SRC␣
↪SUPG
EQ MOMENTUM FOR VELOCITY ON BLOCK_1_POS USING Q1 WITH MASS ADV DIFF SRC␣
↪SUPG
EQ CONTINUITY   FOR PRESSURE   ON BLOCK_1_NEG USING Q1 WITH DIV
EQ CONTINUITY_2 FOR PRESSURE_2 ON BLOCK_1_POS   USING Q1 WITH DIV
```

In the above command block, `MOMENTUM` equation systems are defined on both CDFEM-decomposed blocks `BLOCK_1_POS` and `BLOCK_1_NEG`. The `CONTINUITY` equation system is defined differently; there are 2 distinct pressure fields in the CDFEM domain (`PRESSURE` and `PRESSURE_2`). This is to capture the sharp discontinuity in the pressure field across the interface to balance the capillary force.

The approach for modeling capillary forces in CDFEM is different than a diffuse interface approach. This stems from the fact that we have a surface entity upon which we can enforce boundary conditions (fluxes) on; because of this, we are able to model the capillary forces as a boundary force $\boldsymbol{f_\sigma}$ across the interface instead of a volumetric smoothed force. This is enabled with the following command:

```
BC FLUX FOR MOMENTUM ON SURFACE_BLOCK_1_NEG_POS = CAPILLARY
```

It should be noted that CDFEM provides both `SURFACE_BLOCK_1_NEG_POS` and `SURFACE_BLOCK_1_POS_NEG` in our example; fluxes applied to these surfaces are applied to the `NEG` and `POS` phase/side of the level set interface, respectively. For the above command line, we are applying a capillary flux to the `NEG` phase/side. A G/FEM residual form of this is given as:

$$-\int_S \boldsymbol{f_\sigma} \cdot \boldsymbol{e}_k \phi^i \, dS \tag{4.16}$$

For a `BC FLUX` approach for modeling capillary forces, $\boldsymbol{f_\sigma}$ is computed as:

$$\boldsymbol{f_\sigma} = \sigma \kappa \boldsymbol{n} \tag{4.17}$$

Notice in Eq. (4.17) we no longer need $\frac{\partial H_\varepsilon}{\partial \Phi}$ or ∇H_ε.

An alternative approach to modeling the capillary force is offered through the `FLUXBP` command (which stands for "flux by parts"). This models the capillary component to the `MOMENTUM` equation as a stress tensor $\boldsymbol{T_\sigma}$ as opposed to a surface force vector. The contribution to the G/FEM residual form becomes:

$$\int_S \boldsymbol{T_\sigma} : \boldsymbol{\nabla} \left(\boldsymbol{e}_k \phi^i \right) dS \tag{4.18}$$

The form of $\boldsymbol{T}_{\sigma}$ is similar to the form proposed by LaFaurie et al. ([41]), which takes a Laplace-Beltrami implementation of the capillary force:

$$\boldsymbol{T}_{\sigma} = \sigma \left(\boldsymbol{I} - \boldsymbol{n} \otimes \boldsymbol{n} \right) \tag{4.19}$$

where $\boldsymbol{I}$ is the identity tensor. Notice that there is no curvature term κ in Eq. (4.19) like there is in Eq. (4.17); this avoids the need to compute a second order derivative on the level set field Φ. A detailed explanation of this $\boldsymbol{T}_{\sigma}$ term is given by Cairncross et al. [42].

An additional stabilization term $\boldsymbol{T}_{\sigma}^{stab}$ can be added to Eq. (4.19) through the `FLUXBP` option. This acts to stabilize the capillary forces against larger time steps, and follows the form proposed by Hysing et al. ([43]):

$$\boldsymbol{T}_{\sigma}^{stab} = -\mu \left(\boldsymbol{n} \cdot \nabla \boldsymbol{v} \right) \otimes \boldsymbol{n} \tag{4.20}$$

where μ is the fluid viscosity. This term can be added in the input deck as

```
BC FLUX FOR MOMENTUM ON SURFACE_BLOCK_1_NEG_POS = CAPILLARY_STABILIZATION
```

Element Death

Element death is a dynamic process in which existing elements are identified by their state as being active (included in the simulation) or inactive/dead (excluded from the simulation). This capability is implemented in two different flavors, one where elements are explicitly removed (Standard Element Death) and another (CDFEM) in which elements are subdivided in a manner consistent with the element death criterion. Tradeoffs between the two include:

- CDFEM has been shown to possess a notion of solution convergence, whereas standard element death is only qualitatively correct.
- Standard element death can be applied to bulk volume discretizations of any element topology, whereas CDFEM element death is limited to 2D triangular and 3D tetrahedral discretizations.
- Generally speaking, CDFEM is more computationally expensive than standard element death but provides a higher resolution in front tracking.

With a few exceptions both element death implementations utilize essentially the same command directives.

The element death capability is composed of two parts, definition of criterion and definition of its scope. Each death criterion definition must be named uniquely (`DeathName`) and its scope will be the specific set of mesh entities, blocks of the input mesh to which the criterion is applied. See the *command reference* for details on all available commands.

Attention: While an element death definition can include many mesh entities, these entities must correspond to a single element topology, i.e. Hex8, Tet4.

The element death process will result in a new discretization that may include surfaces defined by dead elements. The new discretization will be written to a results file named the same as the original results file but appended with a suffix, e.g. for an original file `t.e` the new discretization would be `t.e-s0002`. For simulations in which elements are continually being killed this results in a numbered sequence of output files.

Note: Visualization of element death differs between standard element death and CDFEM element death but is somewhat straightforward as most applications will automatically load the collection of output files to produce the appropriate model animation.

Standard Element Death

In standard element death, characterization of the death state is monitored using an element variable (`DEATH_STATUS`) which denotes whether an element is currently active (`DEATH_STATUS`= 1) or inactive (`DEATH_STATUS`= 0). Visualization of a model with the "dead" elements removed is made possible by use of the `DEATH_STATUS` output variable. Since `DEATH_STATUS`= 0 denotes dead elements, the `DEATH_STATUS` variable can then be used in conjunction with any visualization thresholding utility to animate the element death process.

During a simulation in which standard element death is defined, one can monitor the extent of element death through two values output to the log file, `KILLED_ELEMENTS` and `TOTAL_DEAD_ELEMENTS`. `KILLED_ELEMENTS` reports the number of elements removed from the simulation after completion of the current time step while `TOTAL_DEAD_ELEMENTS` is the cumulative number of elements deleted. Both `KILLED_ELEMENTS` and `TOTAL_DEAD_ELEMENTS` are also available for *output* to the results/heartbeat output as global variables under the same name (e.g. `global variables = killed_elements`).

One can also monitor the extent of element death by observing the number of unknowns in the nonlinear solution step portion of the log file. More specifically, the content of the line beginning with `* Matrix:Solver:` which contains the number of unknowns on processor rank 0 but and the following line beginning with `* Mesh` will contain the number of elements on processor rank 0.

CDFEM Element Death

In CDFEM, characterization of the death state is monitored directly by the contents of two element block entities: the original element block and an internally generated "death" part. The "death" part has the base name of original element block appended with `_DEAD` (e.g. the "death part" for `BLOCK_3` would be `BLOCK_3_DEAD`). When element death occurs, elements in the original element block are removed and transitioned to the "death" part. Since the "death" part is not part of the original mesh, the user must define this part within the *Finite Element Model* command block in the same way as the original element block was defined e.g.

```
# Scope: Sierra
Begin Finite Element Model The_Mesh
  Database Name = mesh.g
  Decomposition Method = rcb

  use material foo for block_1
  use material foo for block_1_dead
  ...
End
```

During a simulation in which CDFEM element death is active, one can roughly monitor the extent of element death by observing the number of unknowns in the nonlinear solution step portion of the log file.

Note: As state above, the content of the line beginning with `* Matrix:Solver:` contains the number of unknowns on processor 0, but changes on processors other than rank 0 are not recorded in the log file.

Visualization of a model with CDFEM element death is straightforward since a clear distinction is made between the element blocks in the original mesh and the "death" part. Hence one can proceed in the same way as the original mesh.

General Death Syntax

Death criteria can be expressed in terms of any internal field. As an example, a standard element death command block named criteria `DEATH_TEMP8` based upon the average nodal temperature solution on `BLOCK_3` exceeding temperature of 322 K might be defined as:

```
# Scope: Sierra > Procedure > Aria Region
Begin Element Death death_temp8
  add volume block_3
  Criterion is Avg nodal value of solution->Temperature > 322.0
End
```

The equivalent syntax for CDFEM death differs only in the command block specification, **Begin** CDFEM death death_temp8.

```
# Scope: Sierra > Procedure > Aria Region
Begin CDFEM Death death_temp8
  add volume block_3
  Criterion is Avg nodal value of solution->Temperature > 322.0
End
```

As elements are deactivated, elements that were previously internal can become exposed as external surfaces. For consistent applications of boundary conditions, the interface between the active and inactive elements is captured in a surface part named *surface_<DEATHNAME>* e.g. surface_death_temp8. As an example, with standard element death a flux boundary condition applied to the surface of a block with element death occurring can be achieved with

```
# Scope: Sierra > Procedure > Aria Region
Begin Heat Flux Boundary Condition X-face
  Add Surface surface_4
  Add Surface surface_death_temp8
  Flux = 100.
End
```

where the addition of Add Surface surface_death_temp8 ensures that the boundary condition surface is continually updated.

Note: Element death impacts the existence of the element in all physics defined in the Aria Region. In problems defined with segregated physics, then, the element death behavior must be defined outside the scope of any Equation System.

In most cases the element death criteria will be evaluated based upon values computed within Aria. Here the death criteria are evaluated at the end of a solution step. With standard element death, it is also possible to evaluate the criteria based upon externally supplied "transfer" variables. In this case one can force the death criteria to be evaluated at the beginning of time step using the *Transfer Element Death* command line.

Burn Front Model

Some energetics are modeled in Aria using finite rate chemical mechanisms; however, this approach can be more expensive than is necessary to capture some thermal effects. An alternate approach is to define a burn front using a level set, which moves at a prescribed outward velocity and releases heat as it moves.

To enable this capability, the burn front model provides a mechanism for initializing the level set field based on an initiation or ignition temperature, or tied to chemeq deactivation based on the

chemeq deactivation criteria (either temperature or temperature rate).

Model Setup

To set up a burn front model, you must make the following changes to the input file:

- Define the burn model parameters (*burn speed*, *heat release*, *initiation*, *ignition*, and *front width*) in each energetic material. The full set of available models for each property is listed in the linked command reference sections.

```
Burn Speed           = Constant     Value = 0.01   # m/s
Burn Heat Release    = Constant     Value = 1e6    # J/kg
Burn Initiation      = Temperature  Tign  = 750    # K
Burn Front Width     = Constant     Width = 0.005  # m
```

- Add a level set equation before your temperature equation.

```
Begin Equation System LevelSet
  [SOLVER SETTINGS]

  EQ Level_Set for Level_Set on all_EM Using Q1 with mass ADV
  EQ Extension_Speed for Extension_Speed on all_EM Using Q1 with xfer
End Equation System LevelSet
```

- Add an energy source term from the level set burn.

```
Source for energy on block_1 = Burn_Front
```

- Set the initial condition for the level set field to a large positive number everywhere (needs to be larger than the specified level set width).

Model Assumptions and Limitations

This section contains various notes about how the model will behave under certain conditions, and implicit assumptions and limitations of the model.

- By default each block can only have one ignition event. Once it is ignited it initializes the level set field and does not monitor for subsequent ignition. For multiple ignition events, you must specify a 'min_ignition_spacing = X' value on the Burn Initiation line. This will allow multiple ignition events as long as they are separated from an existing burn front by the specified distance. Pick a minimum ignition spacing (d) that is several times larger than your interface width ($d > 3w$), and that provides $Da \gg 1$, where $Da = \frac{v_b d}{\alpha}$ (α = thermal diffusivity and v_b = burn speed). This is so that the thermal front cannot move ahead of the burn front and trigger non-physical internal ignition points.

```
Burn Initiation = Temperature  Tign = 750  min_ignition_spacing = 0.
↪01
```

- There are two valid forms for burn initiation, i.e., how the level set is initialized given ignition has occurred. The first, detailed above, is by an initiation temperature. The second is by using the initialization criteria for chemeq deactivation. Such a command line would look like

```
Burn Initiation = chemeq
```

- In addition to burn initiation (where the burn starts), burn ignition (when the burn starts) can be specified. When using the chemeq criterion for burn initiation, the ignition is automatically done when chemeq is deactivated, and there is no option to override the ignition criteria. It is possible to use temperature burn initiation in conjunction w/ ignition based on chemeq by using the commands

```
Burn Initiation = Temperature Tign = 750
Burn Ignition = Chemeq
```

- The level set is distanced using a global distance, so a tortuous material may result in a burn front that jumps from one location to a nearby location. Using the fast-marching level set may alleviate this issue, but is only compatible with tet meshes.
- The total energy release from the burn model is independent of time step size and burn front speed, but the local temperature profile may not be. For example, if a heat flux is applied to a block in a burn front model and a small time step is used, the nodes on the heated surface will ignite and the front will propagate into the block. If a very large time step is used, the entire block may ignite at once releasing its heat uniformly. If time-resolving the local temperature increase due to the burn front is important to a particular application then a Courant limit can be applied to control the time step based on the burn front interface speed and local mesh size:

```
Begin Solution Control Description
   Use System Main
   Begin System Main
      ...
      Begin Transient TimeBlock
         Advance The_Region
      End
   End
   Begin Parameters For Transient TimeBlock
      ...
      Begin Parameters For Aria Region The_Region
         ...
         Time Step Variation     = Adaptive
```

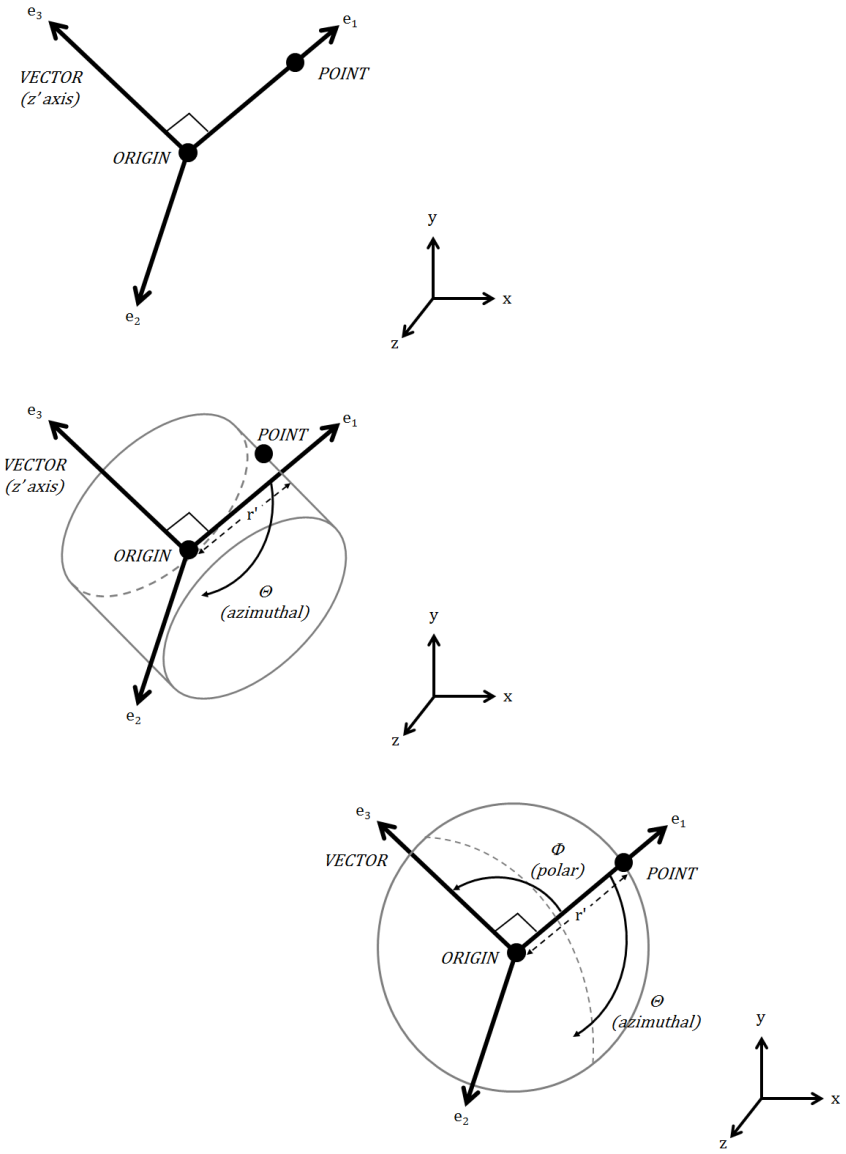
(continues on next page)

(continued from previous page)

```
          Interface Courant Limit = 1.0
          ...
       End
    End
End
```

Local Coordinate System

Local cartesian, cylindrical, and spherical coordinates systems are available in Aria. The figure below shows each local coordinate system definition.



Local cartesian, cylindrical, and spherical coordinate systems.

Local coordinate systems are particularly useful for defining tensor material properties in terms of their local principal values. The input deck below shows the incremental change needed to define a `tensor thermal conductivity` for a `generalized` heat conduction model using a local cartesian coordinate system.

```
Begin Sierra myJob

  ...

  Begin Aria Material mat_cartesian
    ...
    # Define principal values of tensor thermal conductivity
    tensor thermal conductivity = constant T11=100.0 T22=45.0 T33=20.0
    heat conduction = generalized
  End

  # Define a local cartesian coordinate system
  Begin local coordinate system my_cartesian
    TYPE   = cartesian
    ORIGIN =  0.0  0.0  0.0
    POINT  =  5.0  0.0  0.0
    VECTOR =  0.0  0.0  1.0
  End

  Begin Finite Element Model cube
    ...
    coordinate system is cartesian

    Begin parameters for block block_2
      material mat_cartesian

      # Define a local cartesian coordinate system only for block_2
      local coordinate system = my_cartesian
    End
  End
  ...
End
```

Here, the line command `local coordinate system` in the parameters block refers to a local coordinate system command block. This command block defines the `TYPE` of coordinate system (i.e., cartesian, cylindrical, or spherical), `ORIGIN` of the coordinate system, as well as a `VECTOR`, and `POINT`. Together, this information is used to define the principal directions $\mathbf{e}_1$, $\mathbf{e}_2$ and $\mathbf{e}_3$ as is discussed below.

In the material block, the `tensor thermal conductivity` material model defines the principal values K_{11}, K_{22} and K_{33} corresponding to principal directions $\mathbf{e}_1$, $\mathbf{e}_2$ and $\mathbf{e}_3$ respectively. With $\mathbf{K}$ defined locally, the tensor components in global coordinates are defined as

$$\mathbf{K}^{'} = \mathbf{Q}\mathbf{K}\mathbf{Q}^{T} \tag{4.21}$$

where the transformation matrix $\mathbf{Q}$ is defined as

$$\mathbf{Q}_{ij} = \mathbf{e}_i^{'} \cdot \mathbf{e}_j. \tag{4.22}$$

Here, $\mathbf{e}_i^{'}$ is the global cartesian basis, resulting in a transformation matrix $\mathbf{Q}$ whose columns are the principal directions $\mathbf{e}_j$. Physically, $\mathbf{Q}$ transforms local vector coordinates to global vectors coordinates, whereas $\mathbf{Q}^T$ transforms global vector coordinates to local vector coordinates.

The line commands `ORIGIN`, `POINT`, and `VECTOR` define vectors $\mathbf{o}$, $\mathbf{p}$, and $\mathbf{v}$ respectively. For the cartesian coordinate system, the principal directions do not vary with position and are defined as

$$\mathbf{e}_1 = \frac{\mathbf{d}}{||\mathbf{d}||_2} \qquad \mathbf{e}_2 = \mathbf{e}_3 \times \mathbf{e}_1 \qquad \mathbf{e}_3 = \frac{\mathbf{v}}{||\mathbf{v}||_2}, \tag{4.23}$$

where $\mathbf{d} = \left(\mathbf{I} - \mathbf{e}_3\mathbf{e}_3^T\right) \cdot (\mathbf{p} - \mathbf{o})$ is independent of position. Here, `VECTOR` defines the local z-axis and `POINT` determines the local x-axis such that it is orthogonal to the local z-axis.

For the cylindrical and spherical coordinate systems, the principal directions vary from point to point within a material block. At a given point $\mathbf{r}$, the cylindrical principal directions are defined as:

$$\mathbf{e}_1 = \frac{\mathbf{d}\,(\mathbf{r})}{||\mathbf{d}\,(\mathbf{r})\,||_2} \qquad \mathbf{e}_2 = \mathbf{e}_3 \times \mathbf{e}_1 \qquad \mathbf{e}_3 = \frac{\mathbf{v}}{||\mathbf{v}||_2}, \tag{4.24}$$

where $\mathbf{d}\,(\mathbf{r}) = \left(\mathbf{I} - \mathbf{e}_3\mathbf{e}_3^T\right) \cdot (\mathbf{r} - \mathbf{o})$ depends on the position now. `VECTOR` defines the local z-axis of the cylinder and so these principal directions correspond to r, θ, and z directions of a cylinder, respectively.

Similarly, at a given point $\mathbf{r}$, the spherical principal directions are defined as:

$$\mathbf{e}_1 = \frac{\mathbf{r} - \mathbf{o}}{||\mathbf{r} - \mathbf{o}||_2} \qquad \mathbf{e}_2 = \mathbf{v} \times \mathbf{e}_1 \qquad \mathbf{e}_3 = \mathbf{e}_1 \times \mathbf{e}_2. \tag{4.25}$$

Here, the principal directions correspond to r, θ, and ϕ directions of the sphere. The command reference for the local coordinate system can be found *here*.

Correlation Heat Transfer Coefficient

Table of Contents

- *Correlations Listing*
- *Forced Convection (Internal)*
- *Forced Convection (External)*
- *Free Convection (External)*

A convective flux boundary condition $q_n = h\left(T - T_{fluid}\right)$ can be used to model problems where a full conjugate heat transfer simulation is deemed impractical. To facilitate specification of these models, Aria includes a number of heat transfer coefficient correlations [44] that cover laminar and turbulent flows of forced and free convection of gas and liquids (including liquid metals) for internal and external flow geometries.

All correlations define a Nusselt number Nu from which the heat transfer coefficient h is computed as

$$h = \frac{k\mathrm{Nu}}{D_h}, \tag{4.26}$$

where k is the scalar thermal conductivity and D_h is a characteristic length for the problem. As shown in the example below, correlations are defined in `heat transfer correlation coefficient` command blocks and are referenced in the `convective flux boundary condition` command block.

```
Begin Sierra myJob
  ...
  Begin Aria Material my_fluid
    ...
    # Define k, rho, cp, mu models
    thermal conductivity = constant value = 1.0
    specific heat capacity = constant value = 1000.0
    density = constant value = 10.0
    viscosity = constant value = 0.01
  End
  ...

  Begin aria region my_region
    ...
    Begin equation system energy
      ...

      # Define conv flux BC using specified correlation
      Begin convective flux boundary condition bar_coupling
        add surface surface_1
```

(continues on next page)

(continued from previous page)

```
        use correlation convection model pipe_flow
        use advective bar my_bar
      End

      # Equivalent line cmd
      # BC Bar_Flux for energy on surface_1 = generalized_nat_conv bar =␣
↪my_bar hcorr = pipe_flow

      # Define h
      Begin heat transfer correlation coefficient pipe_flow
        laminar correlation = type 2 # Nu = 3.666 analytic laminar pipe␣
↪flow
        turbulent correlation = type 72 # Gnielisnki correlation w/ Nu =␣
↪func(Re, Pr, f)

        # Define Reynolds number above which the turbulent correlation␣
↪will be used
        transition Reynolds Number = 3000

        # Define additional inputs needed to compute Nu
        compute characteristic length model = hydraulic diameter
        compute friction factor model = smooth tube
        compute fluid velocity model = advection_velocity
        compute reynolds number
        compute prandtl number
      End

      # Postprocess average htc
      postprocess average of expression heat_transfer_coefficient on␣
↪surface_1 as avg_htc
    End
  End
End
```

In this example, laminar and turbulent correlations are used to model the conjugate heat transfer of internal flow past a pipe. To place emphasis on the correlation, details concerning the *advective bar* fluid model or pipe heat conduction model have been omitted.

To start, all heat transfer correlation models must define the thermal conductivity k and characteristic length D_h. The thermal conductivity is always specified as a material property, whereas the characteristic length is specified via a line command. In this example, the `hydraulic diameter` model (as specified in the advective bar block) would be used for the characteristic length.

In the above example, the type 72 Gnielinski correlation has been used. For this correlation, the Nusselt number is a function of the Reynolds number $\mathrm{Re} = \frac{UD_h}{\nu}$, Prandtl number $\mathrm{Pr} = \frac{\nu}{\alpha}$, and non-dimensional friction factor. Correspondingly, line commands need to be specified to compute each of these non-dimensional parameters. Here, for the friction factor computation, the `smooth tube` model has been specified whereas the Reynolds number and Prandtl number will be computed from material properties defined in `aria material` command block and applied to the volume bar block. Note a characteristic speed also needs to be specified to compute the Reynolds number. Here, the `compute fluid velocity` line command specifies to use the `advection_velocity` expression.

While this example has highlighted how to define and use a forced convection correlation with $T_{fluid}(\mathbf{x}, t)$ modeled using a 1D *advective bar*, in a similar manner free convection correlations can be defined with the bulk fluid temperature $T_{fluid}(t)$ modeled using a *bulk node*.

Below is an example of using an external free convection correlation with a bulk node. To start, we see the `use advective bar` line command has been replaced with the `use bulk element` line command, where the specified name now refers to a bulk node. Next, for the correlation, Nu is now a function of Pr and the Ra Rayleigh number. As this free convection correlation does not depend on a friction factor, or Reynolds number, there is no need to specify them. This includes any unused prerequisites (e.g. characteristic speed). Unlike before, the Rayleigh number is automatically computed for the given correlation from its prerequisites and so no line command is needed. However, the Rayleigh number prerequisites do need to be specified. Note that material properties are now pulled from the material assigned to the bulk node.

```
Begin Sierra myJob
  ...

  # Define k, rho, cp, mu models
  Begin Aria Material my_bulk_material
    viscosity            = user_function name = air_viscosity X =␣
↪temperature
    density              = user_function name = air_density   X =␣
↪temperature
    specific heat        = user_function name = air_specheat  X =␣
↪temperature
    thermal conductivity = user_function name = air_cond      X =␣
↪temperature
  End
  ...

  Begin aria region my_region
    ...

    # Define bulk node
```

(continues on next page)

(continued from previous page)

```
    Begin bulk fluid element my_bulk_node
      ...
      material = my_bulk_material
      Initial Temperature = 600.0

      # Setting a large volume here will in effect result in an infinite␣
↪reservoir
      bulk element volume = constant v = 1.0e10
      bulk eq energy for temperature using p0 with mass src
    End

    Begin equation system energy
      ...

      # Define conv flux BC using specified correlation
      Begin convective flux boundary condition bulk_node_coupling
        add surface surface_1
        use correlation convection model external_free_conv
        use bulk element my_bulk_node
      End

      # Define h
      Begin heat transfer correlation coefficient external_free_conv
         LAMINAR CORRELATION = type 64
         TURBULENT CORRELATION = type 64 # Churchill-Chu free convection

         # Define D_h as a constant instead of using an expression
         characteristic length = 1.0

         # Define additional inputs needed to compute Nu(Ra, Pr)
         gravitational constant = 9.8 # g
         wall length = 1.0 # L
         compute prandtl number # Pr
         compute wall temperature # T_w
         compute fluid temperature # T_f
      End

      # Postprocess average htc
      postprocess average of expression heat_transfer_coefficient on␣
↪surface_1 as avg_htc
    End
  End
End
```

Note: In some cases, it is desirable to use one of the built in heat transfer correlations with a fixed reference temperature. The current work around is to couple the surface to a bulk node with a large volume assigned in order to mimic an infinite reservoir. The reference temperature will fluctuate near the initial temperature conditions applied to the bulk node.

Warning: Both laminar and turbulent correlations are always required to be specified. When both correlations are free convection correlations, default constant expressions of one are set for the transitional Reynolds number, characteristic speed, and Reynolds number (if not already provided). This behavior will be fixed in the future so that specification of both correlations is not always required. For now, setting both correlations to the same type can be used as a work around.

As shown in these examples, depending on the correlation, additional line commands will need to be specified. The supported line commands can be found *here*. If a correlation prerequisite is not met, an error will be thrown indicating the prerequisite expressions that the correlation needs. Supported correlations are listed next.

Correlations Listing

The following parameters will be referred to in the various correlation definitions:

Table 4.1: Parameter Definitions

Symbol	Name
f	friction factor
Re	flow Reynolds number
Pr	flow Prandtl number
Ra	flow Rayleigh number
Pr_w	flow Prandtl number evaluated with wall conditions
T	fluid temperature
T_w	wall temperature
L	wall length
g	gravitational constant
r	ratio of annulus inner diameter to outer diameter
ρ	fluid density
μ	dynamic viscosity
ν	kinematic viscosity
β	expansion coefficient
θ	wall angle (in radians)
D_e	entrance length

Forced Convection (Internal)

Type 1

This correlation applies to cases of fully-developed internal laminar flow with a constant wall flux or constant wall temperature, for which a Nusselt number may be determined analytically.

$$\mathrm{Nu} = 4.36 \tag{4.27}$$

Regime

Laminar

Phase

Gas|Liquid

Range

$\mathrm{Pr} > 0.6$

Type 2

This correlation applies to cases of fully-developed internal laminar flow with a constant wall flux or constant wall temperature, for which a Nusselt number may be determined analytically.

$$\mathrm{Nu} = 3.666 \tag{4.28}$$

Regime

Laminar

Phase

Gas|Liquid

Range

$\mathrm{Pr} > 0.6$

Type 3

This correlation by Hausen applies to cases of thermally fully-developed laminar flow with entrance effects.

$$\begin{aligned} \mathrm{Nu} &= 3.66 + \frac{0.0668 I}{1 + 0.4 I^{0.66667}} \\ I &= \frac{D_h}{D_e} \mathrm{RePr} \end{aligned} \tag{4.29}$$

Regime

Laminar

Phase

Gas|Liquid

Range

$\mathrm{Pr} \gg 1$ or unheated entrance

Type 4

This correlation by Nusselt applies to cases of thermally fully-developed turbulent flow with entrance effects.

$$\mathrm{Nu} = 0.036\mathrm{Re}^{0.8}\mathrm{Pr}^{0.3333}\left(D_h/D_e\right)^{0.05556} \tag{4.30}$$

Regime

Turbulent

Phase

Gas|Liquid

Type 14

This correlation by Azer-Chao applies to cases of constant surface temperature liquid metals in tubes.

$$\mathrm{Nu} = 5.0 + 0.05\left(\mathrm{RePr}\right)^{0.8}\mathrm{Pr}^{0.25} \tag{4.31}$$

Regime

Turbulent

Phase

Liquid Metal

Type 15

This correlation by Sliecher applies to cases of constant surface temperature liquid metals in tubes.

$$\mathrm{Nu} = 4.8 + 0.0156\left(\mathrm{RePr}\right)^{0.85}\mathrm{Pr}^{0.08} \tag{4.32}$$

Regime

Laminar|Turbulent

Phase

Liquid Metal

Range

$0.004 < \mathrm{Pr} < 0.1$ and $\mathrm{Re} < 500000$

Type 23

This correlation by Petukhov applies to cases of turbulent gas flow with friction factor correlation for smooth surfaces.

$$\mathrm{Nu} = \frac{0.125 f RePr}{1.07 + 12.7\sqrt{0.125 f}\left(\mathrm{Pr}^{0.6667} - 1\right)} \tag{4.33}$$

Regime

Turbulent

Phase

Gas

Range

$3000 < \mathrm{Re} < 5000000$ and $0.5 < \mathrm{Pr} < 2000$

Type 25

This correlation by Dittus-Boelter applies to cases of turbulent flow.

$$\begin{aligned} \mathrm{Nu} &= 0.023\mathrm{Re}^{0.8}\mathrm{Pr}^{e} \\ e &= \begin{cases} 0.3 & T_w < T \\ 0.4 & \text{else} \end{cases} \end{aligned} \tag{4.34}$$

Regime

Turbulent

Phase

Gas|Liquid

Range

$10000 < \mathrm{Re} < 1000000$ and $0.7 < \mathrm{Pr} < 160$

Type 26

This correlation by Sieder-Tate applies to cases of turbulent gas flow.

$$\mathrm{Nu} = 0.027\mathrm{Re}^{0.8}\mathrm{Pr}^{0.3333}\left(\frac{T}{T_w}\right)^{0.098} \tag{4.35}$$

Regime

Turbulent

Phase

gas

Range

$10000 < \mathrm{Re} < 1000000$ and $0.7 < \mathrm{Pr} < 160$

Type 39

This correlation by Sieder-Tate applies to cases of turbulent liquid flow.

$$\begin{aligned} \mathrm{Nu} &= 0.027\mathrm{Re}^{0.8}\mathrm{Pr}^{0.3333}\left(\frac{\mu}{\mu_w}\right)^{0.14} \\ \mu_w &= \left(\mu^{-0.2661} + \frac{T_w - T}{37.073}\right)^{-3.758} \end{aligned} \tag{4.36}$$

Regime

Turbulent

Phase

Liquid

Range

$10000 < \mathrm{Re} < 1000000$ and $0.7 < \mathrm{Pr} < 160$

Type 72

This correlation by Gnielinski applies to cases of turbulent gas flow in a circular tube with friction factor correlation for smooth surfaces.

$$\mathrm{Nu} = \frac{\frac{f}{8}(\mathrm{Re} - 1000)\mathrm{Pr}}{1 + 12.7\sqrt{\frac{f}{8}}\left(\mathrm{Pr}^{0.6667} - 1\right)} \tag{4.37}$$

Regime

Turbulent

Phase

Gas

Range

$3000 < \mathrm{Re} < 5000000$ and $0.5 < \mathrm{Pr} < 2000$

Type 73

This modified correlation by Gnielinski applies to cases of turbulent flow in an annulus with friction factor correlation for annular surfaces where the inner wall is heated and the outer wall is insulated.

$$\begin{aligned} \mathrm{Nu} &= cF\frac{\frac{f}{8}\mathrm{RePr}}{b + 12.7\sqrt{\frac{f}{8}}\left(\mathrm{Pr}^{0.6667} - 1\right)} \\ b &= 1.07 + \frac{900}{\mathrm{Re}} - \frac{0.63}{1 + 10Pr} \\ F &= 0.75r^{-0.17} \\ c &= \begin{cases} 1 & \text{GAS and } T > T_w \\ \left(\frac{T}{T_w}\right)^{0.45} & \text{GAS and } T < T_w \\ \left(\frac{\mathrm{Pr}}{\mathrm{Pr}_w}\right)^{0.11} & \text{LIQUID} \end{cases} \end{aligned} \tag{4.38}$$

Regime

Turbulent

Phase

Gas|Liquid

Range

$\mathrm{Re} > 10000$

Type 74

This modified correlation by Gnielinski applies to cases of turbulent flow in an annulus with friction factor correlation for annular surfaces where the inner wall

is insulated and the outer wall is heated.

$$\begin{aligned} \mathrm{Nu} &= cF\frac{\frac{f}{8}\mathrm{RePr}}{b + 12.7\sqrt{\frac{f}{8}}\left(\mathrm{Pr}^{0.6667} - 1\right)} \\ b &= 1.07 + \frac{900}{\mathrm{Re}} - \frac{0.63}{1 + 10Pr} \\ F &= 0.9 - 0.15r^{0.6} \\ c &= \begin{cases} 1 & \text{GAS and } T > T_w \\ \left(\frac{T}{T_w}\right)^{0.45} & \text{GAS and } T < T_w \\ \left(\frac{\mathrm{Pr}}{\mathrm{Pr}_w}\right)^{0.11} & \text{LIQUID} \end{cases} \end{aligned} \tag{4.39}$$

Regime
Turbulent

Phase
Gas|Liquid

Range
Re > 10000

Forced Convection (External)

Type 5

This modified Ranz-Marshall correlation applies to cases of thermally fully-developed flow with entrance effects.

$$\mathrm{Nu} = 2 + 0.6\sqrt{\mathrm{Re}}\mathrm{Pr}^{0.3333}25(D_e/D_h)^{-0.7} \tag{4.40}$$

Region
Laminar|Turbulent

Phase
Liquid

Type 18

This correlation applies to cases of flow over a sphere.

$$\mathrm{Nu} = 0.37\mathrm{Re}^{0.6}1.126\mathrm{Pr}^{0.3333} \tag{4.41}$$

Regime
Laminar|Turbulent

Phase
Gas

Range
17 < Re < 700000

Type 21

This correlation by Zhulkauskas applies to cases of aligned cylindrical tube bank in cross flow.

$$\begin{aligned} \mathrm{Nu} &= C_1 \mathrm{Re}^{C_2} \mathrm{Pr}^{0.36} \\ (C_1, C_2) &= \begin{cases} (0.27, 0.63) & \mathrm{Re} < 200000 \\ (0.021, 0.84) & \text{else} \end{cases} \end{aligned} \tag{4.42}$$

Regime

Laminar|Turbulent

Phase

Gas

Range

$1 < \mathrm{Re} < 1000000$ and $0.7 < \mathrm{Pr} < 500$

Type 27

This correlation by Whitaker applies to cases of gas flow over a sphere.

$$\mathrm{Nu} = 2 + (0.4\sqrt{\mathrm{Re}} + 0.06\mathrm{Re}^{0.6667})\mathrm{Pr}^{0.4}\left(\frac{T}{T_w}\right)^{0.175} \tag{4.43}$$

Regime

Laminar|Turbulent

Phase

Gas

Range

$3.5 < \mathrm{Re} < 76000$ and $0.71 < \mathrm{Pr} < 380$

Type 28

This correlation by Hilpert applies to cases of cylinder in cross flow.

$$\begin{aligned} \mathrm{Nu} &= C_1 \mathrm{Re}_f^{C_2} \mathrm{Pr}^{0.3333} \\ \mathrm{Re}_f &= \mathrm{Re}\left(\frac{T}{T_f}\right)^{0.7} \\ (C_1, C_2) &= \begin{cases} (0.989, 0.33) & \mathrm{Re}_f < 4 \\ (0.911, 0.385) & \mathrm{Re}_f < 40 \\ (0.683, 0.466) & \mathrm{Re}_f < 4000 \\ (0.193, 0.618) & \mathrm{Re}_f < 40000 \\ (0.027, 0.805) & \text{else} \end{cases} \end{aligned} \tag{4.44}$$

Regime

Laminar|Turbulent

Phase

Gas

Range

$0.4 < \mathrm{Re} < 400000$

Type 34

This correlation by Zhulkauskas applies to cases of cylinder in cross flow

$$\begin{aligned}
\mathrm{Nu} &= C_1 {\mathrm{Re}_f}^{C_2} \mathrm{Pr}^{C_3} \\
\mathrm{Re}_f &= \mathrm{Re}\left(\frac{T}{T_f}\right)^{0.7} \\
T_f &= \frac{T + T_w}{2} \\
(C_1, C_2) &= \begin{cases} (0.989, 0.33) & \mathrm{Re}_f < 4 \\ (0.911, 0.385) & \mathrm{Re}_f < 40 \\ (0.683, 0.466) & \mathrm{Re}_f < 4000 \\ (0.027, 0.805) & \text{else} \end{cases} \\
C_3 &= \begin{cases} 0.36 & \mathrm{Pr} > 10 \\ 0.37 & \text{else} \end{cases}
\end{aligned} \tag{4.45}$$

Regime

Laminar|Turbulent

Phase

Gas

Range

$1 < \mathrm{Re} < 1000000$ and $0.7 < \mathrm{Pr} < 500$

Type 35

This correlation by Churchill-Bernstein applies to cases of cylinder in cross flow.

$$\begin{aligned}
\mathrm{Nu} &= 0.3 + \frac{0.62\sqrt{\mathrm{Re}_f}\mathrm{Pr}^{0.3333}}{(1 + (0.4/\mathrm{Pr})^{0.6667})^{0.25}} (1 + (\mathrm{Re}_f/282000)^{0.625})^{0.8} \\
\mathrm{Re}_f &= \mathrm{Re}\left(\frac{T}{T_f}\right)^{0.7} \\
T_f &= \frac{T + T_w}{2}
\end{aligned} \tag{4.46}$$

Regime

Laminar|Turbulent

Phase

Gas

Range

$\mathrm{RePr} > 0.2$

Type 36

This correlation applies to cases of flow over a flat plate.

$$\begin{aligned} \mathrm{Nu} &= 0.332\sqrt{\mathrm{Re}_f}\mathrm{Pr}^{0.3333} \\ \mathrm{Re}_f &= \mathrm{Re}(T/T_f)^{0.7} \\ T_f &= \frac{T + T_w}{2} \end{aligned} \tag{4.47}$$

Regime

Laminar

Phase

Gas

Range

$\mathrm{Pr} > 0.6$

Type 37

This correlation applies to cases of flow over a flat plate.

$$\begin{aligned} \mathrm{Nu} &= 0.664\sqrt{\mathrm{Re}_f}\mathrm{Pr}^{0.3333} \\ \mathrm{Re}_f &= \mathrm{Re}(T/T_f)^{0.7} \\ T_f &= \frac{T + T_w}{2} \end{aligned} \tag{4.48}$$

Regime

Laminar

Phase

Gas

Range

$\mathrm{Pr} > 0.6$

Type 38

This correlation applies to cases of flow over a flat plate.

$$\begin{aligned} \mathrm{Nu} &= 0.0296{\mathrm{Re}_f}^{0.8}\mathrm{Pr}^{0.3333} \\ \mathrm{Re}_f &= \mathrm{Re}(T/T_f)^{0.7} \\ T_f &= \frac{T + T_w}{2} \end{aligned} \tag{4.49}$$

Regime

Turbulent

Phase

Gas

Range

$0.6 < \mathrm{Pr} < 60$

Free Convection (External)

Type 58

This correlation by Churchill-Chu applies to free convection from a vertical surface, or non-vertical surface if angle is less than 60° (measured from vertical).

$$\begin{aligned}
\mathrm{Nu} &= 0.68 + 0.67\frac{\mathrm{Ra}^{0.25}}{(1 + (0.492/\mathrm{Pr})^{0.565})^{0.4444}} \\
\mathrm{Ra} &= \mathrm{GrPr} \\
\mathrm{Gr} &= gcos(\theta)L^3(\rho_f)^2\frac{|T_w - T|}{T_f\mu_f^2} \\
\mu_f &= \mu(T_f/T)^{0.7} \\
\rho_f &= \rho\frac{T}{T_f} \\
T_f &= \frac{T + T_w}{2}
\end{aligned} \tag{4.50}$$

Regime

Laminar

Phase

Gas

Range

$\mathrm{Ra} < 1e^9$ and $\theta < 60°$

Type 59

This correlation by Churchill-Chu applies to free convection from a vertical surface, or non-vertical surface if angle is less than 60° (measured from vertical).

$$\begin{aligned}
\mathrm{Nu} &= 0.68 + 0.67\frac{\mathrm{Ra}^{0.25}}{(1 + (0.492/\mathrm{Pr}_f)^{0.565})^{0.4444}} \\
\mathrm{Ra} &= \mathrm{GrPr}_f \\
\mathrm{Gr} &= gL^3(\rho)^2\frac{|T_w - T|}{T_f\mu_f^2} \\
\mathrm{Pr}_f &= \mathrm{Pr}\frac{\mu_f}{\mu} \\
\mu_f &= \left(\mu^{-0.2661} + \frac{T_f - T}{37.073}\right)^{-3.758} \\
T_f &= \frac{T + T_w}{2}
\end{aligned} \tag{4.51}$$

Regime

Laminar

Phase

Liquid

Range

$\text{Ra} < 1e^{9}$ and $\theta < 60°$

Type 60

This correlation by Churchill-Chu applies to free convection from a vertical surface.

$$\begin{aligned}
\text{Nu} &= \left(0.825 + \frac{0.387\text{Ra}^{0.16667}}{(1 + (0.492/\text{Pr})^{0.5625})^{0.2963}}\right)^2 \\
\text{Ra} &= \text{GrPr} \\
\text{Gr} &= gL^3(\rho_f)^2\frac{|T_w - T|}{T_f\mu_f^2} \\
\mu_f &= \mu(T_f/T)^{0.7} \\
\rho_f &= \rho\frac{T}{T_f} \\
T_f &= \frac{T + T_w}{2}
\end{aligned} \tag{4.52}$$

Regime

Laminar|Turbulent

Phase

Gas

Type 61

This correlation by McAdams applies to free convection from a horizontal surface; either the upper surface of a heated plate or the lower surface of a cooled plate.

$$\begin{aligned}
\text{Nu} &= \begin{cases} 0.54\text{Ra}^{0.25} & \text{Ra} < 1e^{7} \\ 0.15\text{Ra}^{0.3333} & \text{else} \end{cases} \\
\text{Ra} &= \text{GrPr} \\
\text{Gr} &= gL^3(\rho_f)^2\frac{|T_w - T|}{T_f\mu_f^2} \\
\mu_f &= \mu(T_f/T)^{0.7} \\
\rho_f &= \rho\frac{T}{T_f} \\
T_f &= \frac{T + T_w}{2}
\end{aligned} \tag{4.53}$$

Regime

Laminar|Turbulent

Phase

Gas

Range

$1e^4 < \text{Ra} < 1e^{11}$

Type 62

This correlation by McAdams applies to free convection from a horizontal surface; either the upper surface of a cooled plate or the lower surface of a heated plate.

$$\begin{aligned}
\text{Nu} &= 0.27\text{Ra}^{0.25} \\
\text{Ra} &= \text{GrPr} \\
\text{Gr} &= gL^3(\rho_f)^2\frac{|T_w - T|}{T_f\mu_f^2} \\
\mu_f &= \mu(T_f/T)^{0.7} \\
\rho_f &= \rho\frac{T}{T_f} \\
T_f &= \frac{T + T_w}{2}
\end{aligned} \tag{4.54}$$

Regime

Laminar|Turbulent

Phase

Gas

Range

$1e^5 < \text{Ra} < 1e^{10}$

Type 63

This correlation by Morgan applies to free convection from a horizontal cylinder.

$$
\begin{aligned}
\mathrm{Nu} &= C_1 \mathrm{Ra}^{C_2} \\
\mathrm{Ra} &= \mathrm{GrPr} \\
\mathrm{Gr} &= gL^3(\rho_f)^2 \frac{|T_w - T|}{T_f \mu_f^2} \\
\mu_f &= \mu(T_f/T)^{0.7} \\
\rho_f &= \rho \frac{T}{T_f} \\
T_f &= \frac{T + T_w}{2} \\
(C_1, C_2) &= \begin{cases} (0.675, 0.058) & \mathrm{Ra} < 1e^{-2} \\ (1.02, 0.148) & \mathrm{Ra} < 100 \\ (0.85, 0.188) & \mathrm{Ra} < 10000 \\ (0.48, 0.25) & \mathrm{Ra} < 1e^{7} \\ (0.125, 0.333) & \text{else} \end{cases}
\end{aligned}
\tag{4.55}
$$

Regime

Laminar|Turbulent

Phase

Gas

Range

$1e^{-10} < \mathrm{Ra} < 1e^{12}$

Type 64

This correlation by Churchill-Chu applies to free convection from a horizontal cylinder.

$$
\begin{aligned}
\mathrm{Nu} &= \left(0.6 + \frac{0.387\mathrm{Ra}^{0.16667}}{(1 + (0.559/\mathrm{Pr})^{0.5625})^{0.2963}}\right)^2 \\
\mathrm{Ra} &= \mathrm{GrPr} \\
\mathrm{Gr} &= gL^3(\rho_f)^2 \frac{|T_w - T|}{T_f \mu_f^2} \\
\mu_f &= \mu(T_f/T)^{0.7} \\
\rho_f &= \rho \frac{T}{T_f} \\
T_f &= \frac{T + T_w}{2}
\end{aligned}
\tag{4.56}
$$

Regime

Laminar|Turbulent

Phase

Gas

Range

$1e^{-5} < \text{Ra} < 1e^{12}$

Type 65

This correlation by Churchill-Chu applies to free convection from a sphere.

$$
\begin{aligned}
\text{Nu} &= 2 + \frac{0.589\text{Ra}^{0.25}}{(1 + (0.469/\text{Pr})^{0.5625})^{0.4444}} \\
\text{Ra} &= \text{GrPr} \\
\text{Gr} &= gL^3(\rho_f)^2\frac{|T_w - T|}{T_f\mu_f^2} \\
\mu_f &= \mu(T_f/T)^{0.7} \\
\rho_f &= \rho\frac{T}{T_f} \\
T_f &= \frac{T + T_w}{2}
\end{aligned}
\tag{4.57}
$$

Regime

Laminar|Turbulent

Phase

Gas

Range

$\text{Ra} > 1e^{11}$ and $\text{Pr} > 0.7$

Chemistry

Chemistry Overview

Aria provides two ways of modeling chemical contributions to the transport equations it solves: *CHEMEQ* and *General Chemistry*. The purpose of this overview is to describe the differences between these approaches and provide guidelines for which one to select for a given application.

Feature Comparison

As a quick reference, a list of feature differences between the two solvers is shown in Table 4.2 below.

CHEMEQ is best suited for evaluating reactions without species transport in order to obtain a source term for the energy equation. It also links to *pressurization zones* to enable reduced-cost pressurization calculations. CHEMEQ stores its species concentrations in an element field and updates that element field in time while providing a source term for the energy equation.

On the other hand, General Chemistry is best suited for situations where you want to resolve species transport and reaction. General Chemistry is designed only to provide source terms, so there must be a linear system set up for each DOF (energy, species concentrations, etc. . .) to actually update its values in time. For cases where the species are not transported, the added cost of setting up this linear system makes CHEMEQ the more efficient choice.

Table 4.2: Feature table for CHEMEQ and General Chemistry.

Feature	CHEMEQ	General Chemistry
Reaction Blocks	Yes	Yes
Multiphase reactions	No	Yes
Coupled Solve	No	Optional
Provides Species Source	No	Yes
Reversible Reactions	No	Yes
Equilibrium Chemistry	No	Yes

Reaction Summary & Diagnostics

For both modeling options, additional debugging information can be provided by activating the `--arialog chemistry` option from the command line launch command. With this logging activated, Aria will write files containing information on the initial conditions prior to ODE failure at each evaluation point of each element on each processor. The files will be named in the following format: `chemistry.E[eid]-[point].P[proc].txt` for General Chemistry, and `chemeq.[model].E[eid]-[point].P[proc].txt` for CHEMEQ.

A summary of all the reactions parsed from the input is also printed to the log file so you can verify they are evaluating as you have intended. An example of this output is shown below. Details about these terms are described in the *reaction blocks* section.

```
+-------------------------------------------------------------------------
↪----+
| Reaction Summary                                                       ␣
↪    |
+-------------------------------------------------------------------------
↪----+
```

(continues on next page)

(continued from previous page)

```
 Name:                       R1
  Reaction String:           A->B
  Stoich Coeffs:             -1.00000 1.00000
  Temperature Phase:         No Material Phase
  Pressure Phase:            No Material Phase
  Heat of Reaction:          -1.00000e+06
  Multiplier Function:       Unity
  Rate Function:             Arrhenius
      Arguments:             A = 0.100000 Ea = 350.000 beta = 0.00000 R = 1.
↪00000
  Pressure Function:         Unity
  Concentration Function:    Standard
      Arguments:             mu = 1.00000 0.00000
```

CHEMEQ Reference

A CHEMEQ reaction mechanism is specified in a nested block inside an `Aria Material`. The CHEMEQ block should generally define the following things:

- Species names involved in the mechanism
- Reactions in the mechanism
- Species phases (only has an effect when integrating with a *pressurization zone*)
- Concentration/energy release units

```
Begin Aria Material ReactingFoo

  # ...

  BEGIN PARAMETERS FOR CHEMEQ MODEL Foo
    species names  are Epoxy     CO2  CH4
    species phases are Condensed Gas  Gas
    Energy Release Units are Per Unit Mass

    Begin Reaction R1
      Reaction is Epoxy -> CH4 + CO2
      Rate Function = Arrhenius A = 1e9 Ea = 25000 R = 8.314
      Concentration Function = Standard mu = automatic
      Heat of Reaction = -1e8
    End
```

(continues on next page)

(continued from previous page)

```
    Concentration Units = Mass Fractions
  END
End
```

The CHEMEQ model creates an ODE at each integration point that is solved to advance species concentrations and used to calculate an energy source term. The CHEMEQ model is not solved by default unless you include it in an energy source term. Using the `chemeq_heating` source term automatically locates the CHEMEQ model on the specified block and activates it.

```
EQ Energy for temperature on block_1 using Q1 with Mass Diff Src
Source for Energy on block_1 = chemeq_heating
```

Given a reaction rate ω_r and stoichiometric coefficient $\nu_{i,r}$ for species i in reaction r (negative for reactants, positive for products), the time rate of change of the concentration of species i (c_i) is

$$\frac{dc_i}{dt} = \sum_r \omega_r \nu_{i,r}$$

At each nonlinear iteration, CHEMEQ solves this ODE from t^n to t^{n+1} at the current temperature to determine the total heat release rate, $\dot{q}$, which is calculated using the start and end point of this integration and the per-species formation enthalpies ($H_{f,i}$). Optionally, this may be multiplied by density depending on your *reaction units*.

$$\dot{q} = \sum_i H_{f,i} \frac{c_i^{n+1} - c_i^n}{\Delta t}$$

For a complete list of CHEMEQ commands, see the *CHEMEQ command reference*.

Defining Reactions

Reactions are defined in one or more `Begin Reaction` blocks inside the CHEMEQ model block. There is a legacy syntax for specifying reactions by supplying matrices of coefficients (described in *Legacy Reaction Specification*), but that approach is no longer recommended.

Each reaction is defined by an overall rate that is the product of one or more terms, a stoichiometric relationship, and a heat release. The syntax for specifying these reaction blocks is described in *Reaction Blocks*.

CHEMEQ Solver & Initial Conditions

Each CHEMEQ model requires a `CHEMEQ Solver` block that refers to it by name. For example, for the CHEMEQ model `Foo` shown earlier you would need to provide the following solver block in the Aria region scope

```
# Sierra > Aria Procedure > Aria Region
Begin CHEMEQ Solver for Foo
  ODE SOLVER = CVODE ADAMS 12 FUNCTIONAL
  species Epoxy = 1.0
  species CO2   = 0.0
  species CH4   = 0.0
End
```

The `CHEMEQ Solver` block specifies the *ODE solver* to use and the initial conditions for all species in the specified mechanism. For most mechanisms `CVODE ADAMS 12 FUNCTIONAL` is a good option. If the chemistry model is particularly stiff, using `CVODE BDF 5 NEWTON` may be more robust, but it is also slightly slower.

The initial concentrations provided must be in the same units assumed in the CHEMEQ model - Aria does not do any unit conversions for these.

Activation & Deactivation

The `CHEMEQ Solver` block can also be used to specify activation and deactivation conditions for the CHEMEQ mechanism (to have it turn on or off at specific temperature, temperature change rates or a more generic criteria).

```
# Sierra > Aria Procedure > Aria Region
Begin CHEMEQ Solver for Foo
  ODE SOLVER = CVODE ADAMS 12 FUNCTIONAL
  species Epoxy = 1.0
  species CO2   = 0.0
  species CH4   = 0.0

  Activation temperature = 350.0
  Deactivation Temperature = 600.0 terminate
End
```

Once the temperature hits the specified deactivation condition, the CHEMEQ solve stops and the simulation either stops or continues, depending on the option selected. The deactivation mechanism looks for a nodal value that reaches the specified deactivation condition (temperature, temperature rate or a generic condition). An example for a generic condition is to measure the reaction rate (which is correlated with the stiffness of the underlying chemistry ODE). The following steps show how to set this generic condition up:

- The Aria material model that contains the targeted CHEMEQ reaction also needs to define a user expression for deactivation. Because we plan to use the reaction rate, a SPECIES_PRODUCTION model needs to be declared to define the time rate of change of the targeted species:

```
Begin Aria Material EM1
  ...
  #This user expression is needed for the generic deactivation condition
  user expression = scalar_string_function user_tag = "deactivation_fcn"␣
↪f = "abs(epoxy_reaction_rate) > 1e-3"
  ...
  #This species_production model is only needed to define our reaction␣
↪rate
  SPECIES_PRODUCTION of Epoxy = From_Time_Rate_Of_Change
  ...
End
```

- In order to make `epoxy_reaction_rate` a usable expression, we need to add a postprocess command within the equation system (or Aria region)

```
Begin aria region myRegion
  ...
  Begin equation system Energy
    ...
    Postprocess species_production of Epoxy on all_blocks as epoxy_
↪reaction_rate
    ...
  End
  ...
End
```

- Now that `epoxy_reaction_rate` is properly defined for the user expression `deactivation_fcn`, we can use this within our CHEMEQ Solver block to define a generic deactivation condition. Notice that in the following block, we define 3 methods of deactivation. This is valid, and the first criteria to be reached will deactivate the CHEMEQ model:

```
# Sierra > Aria Procedure > Aria Region
Begin CHEMEQ Solver for Foo
  ODE SOLVER = CVODE ADAMS 12 FUNCTIONAL
  species Epoxy = 1.0
  species CO2   = 0.0
  species CH4   = 0.0

  Activation temperature = 350.0
```

(continues on next page)

(continued from previous page)

```
  Deactivation Temperature = 600.0 terminate
  Deactivation Temperature rate = 20 terminate
  Deactivation expression = "deactivation_fcn" terminate
End
```

The deactivation option is often used for energetic materials when the reaction rates start to reach thermal runaway conditions. At that point, the cost of solving the chemistry model becomes prohibitive and it often makes sense to switch to a reduced model for the remainder of the burn. The two common options used for this are *level set burns* and uniform heat release.

To use a uniform heat release after deactivation, you need to specify `Post Deactivation Release Time` and `Post Deactivation Heat Release` to control how long the source should be active for and how much heat to release. When using a *pressurization zone* you should also specify `Post Deactivation Gas Release` (in moles per volume) to produce an appropriate pressurization.

In the example below, there will be a uniform energy source term of 1e9 W/m^3 for 10 seconds after the CHEMEQ model deactivates.

```
Begin CHEMEQ Solver for Foo
  # ...

  Post Deactivation Release Time = 10.0
  Post Deactivation Heat Release = 1e9
  Post Deactivation Gas Release = 1.0

  Deactivation temperature = 550.0 continue

  # ...
End
```

Reaction Units

Some of the burden of managing the units properly for CHEMEQ falls to the user. There are currently two commands required to indicate the units being used in CHEMEQ: `Energy Release Units` and `Concentration Units`.

When you use the `Energy Release Units are Per Unit Mass` command, the total heat release term is multiplied by density (from the material model definition) and is calculated as

$$\dot{q} = \rho \sum_i H_{f,i} \frac{c_i^{n+1} - c_i^n}{\Delta t}$$

where $H_{f,i}$ is formation enthalpy for species i (either provided or computed from reaction heat releases). When you use `Energy Release Units are Per Unit Volume` the energy source

term does not include density, and is calculated as

$$\dot{q} = \sum_i H_{f,i} \frac{c_i^{n+1} - c_i^n}{\Delta t}$$

This choice is directly related to your choice of concentration units in your chemical mechanism. If the concentration units are mass fractions, you should use the `Energy Release Units are Per Unit Mass` option. If the concentration units are mass densities, you should use the `Energy Release Units are Per Unit Volume` option.

There is also a `Concentration Units = X` command that is required in order to extract CHEMEQ concentrations out into material properties. The options for `X` include `mass fractions`, `mole fractions`, `density`, or `molar`.

Post-Processing

Sometimes you may want to extract species concentrations from a CHEMEQ model to use in material properties or to post-process. CHEMEQ creates element fields of its species concentrations that are available for output, but are not easily usable in most Aria post-processors.

The best way to use these quantities is by using the `FROM_CHEMEQ` model to pull them into the appropriate material property in the Aria Material block. You must have specified the concentration units using the `Concentration Units` command in the CHEMEQ model in order to do this. No unit conversions are done, but using an improper conversion will result in an error. For example, if the concentration units are mass fractions, you can only use `FROM_CHEMEQ` with the mass fraction property.

For example, to extract the species mass fractions from CHEMEQ and use them to define thermal conductivity you could do

```
Begin Aria Material ReactingFoo

  # ...

  Species Names = Cs CO2 O2
  Mass Fraction Cs  = From_Chemeq
  Mass Fraction CO2 = From_Chemeq
  Mass Fraction O2  = From_Chemeq

  Thermal Conductivity = Scalar_String_Function f = "4 + 0.1*Cs"

  BEGIN PARAMETERS FOR CHEMEQ MODEL Foo
    species names are Cs CO2 O2
```

(continues on next page)

(continued from previous page)

```
    # ...

    Concentration Units = Mass_Fractions
  END
End
```

To use them in a post-processor after defining them with FROM_CHEMEQ you can use the standard syntax

```
Postprocess average of expression mass_fraction of CO2 on block_1 as avg_
↪co2
```

Legacy Reaction Specification

Warning: This section is provided for reference to help understand and convert legacy CHEMEQ models. We do not recommend using this syntax for new models.

Instead of using reaction blocks, older CHEMEQ models provided a grid of concentration exponents and stoichiometric coefficients along with arrays of activation energies, pre-exponential terms, and heat releases.

For example, the following legacy input

```
BEGIN PARAMETERS FOR CHEMEQ MODEL DEMO
  number of reactions is 3
  species names  are A B C D
  Energy Release Units are Per Unit Mass

  # Define Arrhenius terms and heat release
  Steric Coefficients are         0.0    0.0    0.0
  Log Preexponential Factors are 48.7   37.3   28.1
  Activation Energies are         200000 300000 400000
  Energy Releases are            -1e6    2e7    3e5

  # Define the reaction rate concentration dependence
  Concentration Exponents for A  are 1.0 0.0 0.0
  Concentration Exponents for B  are 0.0 1.0 0.0
  Concentration Exponents for C  are 0.0 0.0 2.0
  Concentration Exponents for D  are 0.0 0.0 0.0
```

(continues on next page)

(continued from previous page)

```
  # Define the reactions
  #  A -> B
  #  B -> C
  #  A + C -> 2D
  Stoichiometric coefficients for A are -1.0  0.0 -1.0
  Stoichiometric coefficients for B are  1.0 -1.0  0.0
  Stoichiometric coefficients for C are  0.0  1.0 -1.0
  Stoichiometric coefficients for D are  0.0  0.0  2.0

  Concentration Units are Mass Fractions
END
```

can be converted to a block style below

```
BEGIN PARAMETERS FOR CHEMEQ MODEL DEMO
  species names  are A B C D
  Energy Release Units are Per Unit Mass

  Begin Reaction R1
    Reaction is A -> B
    Rate Function = Arrhenius A = {exp(48.7)} Ea = 200000 R = 8.314
    Concentration Function = Automatic
    Heat of Reaction = 1e6
  End

  Begin Reaction R2
    Reaction is B -> C
    Rate Function = Arrhenius A = {exp(37.3)} Ea = 300000 R = 8.314
    Concentration Function = Automatic
    Heat of Reaction = -2e7
  End

  Begin Reaction R3
    Reaction is A + C -> 2D
    Rate Function = Arrhenius A = {exp(28.1)} Ea = 400000 R = 8.314
    Concentration Function = C 2.0
    Heat of Reaction = -3e5
  End

  Concentration Units are Mass Fractions
END
```

Note: The sign convention for the legacy heat of reaction is the opposite of the block-style reaction convention.

The legacy style required all reactions to use the same basic rate function, while the block-style format provides significantly greater flexibility.

General Chemistry Reference

Similar to CHEMEQ, a General Chemistry reaction mechanism is specified in a nested block inside an `Aria Material`. The General Chemistry block should generally define the following things:

- Species names involved in the mechanism (should be a subset of the species in the Aria material)
- Reactions in the mechanism
- Species variable name (typically `mass_fraction`, `density`, `bulk_density`, `species`, or `mole_fraction`)

```
Begin Aria Material ReactingFoo

  # ...

  BEGIN general chemistry foo_chem
    Species Names = Epoxy CO2  CH4
    Species Variable Name = mass_fraction

    Begin Reaction R1
      Reaction is Epoxy -> CH4 + CO2
      Rate Function = Arrhenius A = 1e9 Ea = 25000 R = 8.314
      Concentration Function = Standard mu = automatic
      Heat of Reaction = -1e8
    End
  END
End
```

The chemistry source terms can then be included in mass_balance/species and equations using the `CHEMISTRY` or `LUMPED_CHEMISTRY` models:

```
Source for species of Epoxy on block_1 = Chemistry  Mechanism = foo_chem
Source for mass_balance of Epoxy on block_1 = Chemistry  Mechanism = foo_
↪chem
Source for Energy on block_1 = Chemistry  Mechanism = foo_chem
```

For a complete list of General Chemistry commands, see the *General Chemistry command reference*.

Required Prerequisites

Using general chemistry requires that several expressions be defined in the same material block or available on the block it is evaluated on. These expressions, and the conditions when they are required, are described in the following list.

- Specific heat of each species (only required if you request a chemistry source term for the energy equation)
- Enthalpy of each species (only required if you request a chemistry source term for the energy equation, for most of the common specific heat models this is created automatically)
- Temperature
- Pressure (optional, if you do not provide it and use a pressure dependent reaction rate it will use 101325 Pa)
- Concentrations (required, you specify what form of concentration to use in the chemistry block, e.g. mass fraction, density, species, or species fraction)
- Molecular weights of each species (required if you use molar concentrations)

The phases for inputs (temperature and pressure) come from `NO_MATERIAL_PHASE` by default. Non-default values for these can be specified separately for each reaction using the `TEMPERATURE PHASE =` and `PRESSURE PHASE =` commands in the reaction blocks.

Defining Reactions

Reactions are defined in one or more `Begin Reaction` blocks inside the General Chemistry model block. There is a legacy syntax for specifying reactions by supplying matrices of coefficients (described in *Legacy Reaction Specification*), but that approach is no longer recommended.

Each reaction is defined by an overall rate that is the product of one or more terms, a stoichiometric relationship, and a heat release. The syntax for specifying these reaction blocks is described in *Reaction Blocks*.

ODE Solver & Initial Conditions

Unlike CHEMEQ, General Chemistry is integrated directly into standard Aria governing equations for species. This means that initial conditions are supplied using the usual Aria convention (e.g. `IC for mass_fraction of CH4 on block_1 = constant value = 0.0`).

The chemistry ODEs can be solved in two ways using General Chemistry - coupled or segregated.

Coupled - When doing a coupled solve, the reaction rates are included directly into the governing equation source terms and time integration is handled by the overall Aria time integration. This option is more expensive for stiff chemical mechanisms but may be necessary when reaction and transport are tightly coupled. For example, the source for species i with a coupled approach would be:

$$S_{c,i} = \frac{dc_i}{dt} = \sum_r \omega_r \nu_{i,r}$$

```
# Aria Region
Begin Chemistry Solver Parameters For foo_chem
  Chemistry Solver Algorithm = COUPLED
End
```

Segregated - When doing a segregated solve, you must specify an ODE solver to use and the chemistry ODEs are integrated separately from the main Aria time integration. Source terms for species and energy are calculated from the change in state variables during the segregated ODE solve. For example, the source for species i for a segregated approach would be evaluated after the ODE solve to determine c_i^{n+1} as:

$$S_{c,i} = \frac{c_i^{n+1} - c_i^n}{\Delta t}$$

```
# Aria Region
Begin Chemistry Solver Parameters For foo_chem
  Chemistry Solver Algorithm = SEGREGATED
  ODE Solver = CVODE BDF 5 NEWTON
  Absolute Tolerance = 1e-8
  Relative Tolerance = 1e-3
  Maximum Substeps = 100000
End
```

Activation & Deactivation

General Chemistry supports activating and deactivating the chemistry mechanism at specific temperature conditions, similar to CHEMEQ. These commands go in the General Chemistry definition block. After deactivation, you can choose to have the simulation `continue` or `terminate`.

```
Begin Aria Material ReactingFoo
  # ...

  BEGIN general chemistry foo_chem
    # ...

    Activation Temperature = 350
    Deactivation Temperature = 600 continue
  END
End
```

Once the temperature hits the specified deactivation condition, the General Chemistry solve stops and the simulation either stops or continues, depending on the option selected.

The deactivation option is often used for energetic materials when the reaction rates start to reach thermal runaway conditions. At that point, the cost of solving the chemistry model becomes prohibitive and it often makes sense to switch to a reduced model for the remainder of the burn. The two common options used for this are *level set burns* and uniform heat release.

To use a uniform heat release after deactivation, you need to specify `Post Deactivation Release Time` and `Post Deactivation Heat Release` to control how long the source should be active for and how much heat to release. When using a *pressurization zone* you should also specify `Post Deactivation Gas Release` (in moles per volume) to produce an appropriate pressurization.

In the example below, there will be a uniform energy source term of 1e9 W/m^3 for 10 seconds after the General Chemistry model deactivates.

```
BEGIN general chemistry foo_chem
  # ...

  Post Deactivation Release Time = 10.0
  Post Deactivation Heat Release = 1e9
  Post Deactivation Gas Release = 1.0

  Deactivation temperature = 550.0 continue

  # ...
End
```

Reaction Units

There are no unit conversions done for you with General Chemistry. You specify the units of the concentration variables with the `Species Variable Name =` command, and must ensure that the corresponding reaction rate terms are in appropriate units to produce source terms in the correct units (e.g. W/m^3 for energy).

Legacy Reaction Specification

Warning: This section is provided for reference to help understand and convert legacy General Chemistry models. We do not recommend using this syntax for new models.

Instead of using reaction blocks, older General Chemistry models provided a grid of concentration exponents and stoichiometric coefficients along with arrays of activation energies, pre-exponential terms, and heat releases for each reaction.

For example, the following legacy input

```
BEGIN GENERAL CHEMISTRY DEMO
  number of reactions is 3
  species names  are A B C D
  Species Variable Name = mass_fraction

  # Define Arrhenius terms and heat release
  Steric Coefficients     =   0.0    0.0    0.0
  Preexponential Factors  =   1e9    1e10   1e12
  Activation Energies     =   200000 300000 400000
  Heats of Reaction       =  -1e6    2e7    3e5

  # Define the reaction rate concentration dependence
  Concentration Exponents for A  = 1.0 0.0 0.0
  Concentration Exponents for B  = 0.0 1.0 0.0
  Concentration Exponents for C  = 0.0 0.0 2.0
  Concentration Exponents for D  = 0.0 0.0 0.0

  # Define the reactions
  #  A -> B
  #  B -> C
  #  A + C -> 2D
  Stoichiometric coefficients for A = -1.0  0.0 -1.0
  Stoichiometric coefficients for B =  1.0 -1.0  0.0
```

(continues on next page)

(continued from previous page)

```
  Stoichiometric coefficients for C =  0.0  1.0 -1.0
  Stoichiometric coefficients for D =  0.0  0.0  2.0
END
```

can be converted to a block style below

```
BEGIN GENERAL CHEMISTRY DEMO
  species names  are A B C D
  Species Variable Name = mass_fraction

  Begin Reaction R1
    Reaction is A -> B
    Rate Function = Arrhenius A = 1e9 Ea = 200000 R = 8.314
    Concentration Function = Automatic
    Heat of Reaction = -1e6
  End

  Begin Reaction R2
    Reaction is B -> C
    Rate Function = Arrhenius A = 1e10 Ea = 300000 R = 8.314
    Concentration Function = Automatic
    Heat of Reaction = 2e7
  End

  Begin Reaction R3
    Reaction is A + C -> 2D
    Rate Function = Arrhenius A = 1e12 Ea = 400000 R = 8.314
    Concentration Function = C 2.0
    Heat of Reaction = 3e5
  End
END
```

The legacy style required all reactions to use the same basic rate function, while the block-style format provides significantly greater flexibility.

Reaction Blocks

Table of Contents

- *Stoichiometry*
- *Reaction Rates*
  - *Multiplier Functions*
  - *Rate Functions*
  - *Pressure Functions*
  - *Concentration Functions*
- *Heat Release*
- *Reaction Phases*
- *Reversible Reactions*

Both CHEMEQ and General Chemistry use a common syntax for specifying reactions. Each reaction block defines the reaction stoichiometry, the different rate-controlling functions, and optional parameters like the heat of reaction and temperature or pressure phases (not valid for CHEMEQ).

```
Begin Reaction [REACTION NAME]
  Reaction is A + 1.2B -> 3C
  Multiplier Function = [Optional]
  Rate Function = [Required]
  Pressure Function = [Optional]
  Concentration Function = [Optional]
  Heat of Reaction = [Value]
  Temperature Phase = [GAS_PHASE, SOLID_PHASE, NO_MATERIAL_PHASE
↪(default)]
  Pressure Phase = [GAS_PHASE, SOLID_PHASE, NO_MATERIAL_PHASE (default)]
  Origin Phase Heat Fraction = [Value]
End
```

The syntax rules for specifying reactions using these blocks is shown in the following sections.

Stoichiometry

The reaction stoichiometry is defined by the `Reaction is` line. The left hand side and right hand side must be separated by either "->" or "=>" and any numeric constants before a species name is interpreted as a stoichiometric coefficient. When using mass-based units (density or mass fraction) it is required that the stoichiometric coefficient of reactants and products be equal to conserve mass. For molar units (mole fractions or molar concentration) there is no such requirement.

```
Reaction is A + 2B -> C    # not valid for mass-based concentrations
Reaction is A + 2B -> 3C   # valid for mass-based concentrations
```

Reaction Rates

The overall rate of each reaction is given by

$$\omega_r = f_m(t,T,P,area,Y)\, f_r(t,T,P,area,Y)\, f_p(P,T)\, f_c(Y)$$

where f_m is the multiplier function, f_r is the rate function, f_p is the pressure function, and f_c is the concentration function. The available options for each of these functions are outlined in the following section. They can be selected in the input file by adding as many reaction blocks in the parent chemistry description block.

Multiplier Functions

Unity

The default multiplier function is unity,

$$f_m(t,T,P,area,Y) = 1$$

and can be used by either omitting the `Multiplier Function` line or using

```
Multiplier Function = Unity
```

Tanh Time

The `tanh_time` multiplier function uses a hyperbolic tangent function of time to increase the reaction rate at a specified time.

$$f_m(t,T,P,area,Y) = \frac{1}{2}\left[1 + \tanh\left(x_{span}(t - x_{ref})\right)\right](f_{end} - f_{start}) + f_{start}$$

The two inputs for this are x_{ref} and x_{span}. If you do not specify an input for the span, it uses $x_{span} = 1/x_{ref}$, unless x_{ref} was set as 0, in which case an error is thrown. The default start and end points (f_{start} and f_{end}) of the tanh blending function are 0 and 1, respectively. These can be overridden by user-specified start and end values if desired. Example inputs are shown below.

```
Multiplier Function = Tanh_Time x_ref = 54
Multiplier Function = Tanh_Time x_ref = 2.5 span = 0.1
Multiplier Function = Tanh_Time x_ref = 2.5 span = 0.1 start = 1 end = 100
```

Tanh Temperature

The `tanh_temperature` multiplier function uses a hyperbolic tangent function of temperature to increase the reaction rate at a specified temperature.

$$f_m(t, T, P, area, Y) = \frac{1}{2}\left[1 + \tanh\left(x_{span}(T - x_{ref})\right)\right](f_{end} - f_{start}) + f_{start}$$

The arguments for this function as the same format as for the tanh_time function.

```
Multiplier Function = Tanh_Temperature x_ref = 450
Multiplier Function = Tanh_Temperature x_ref = 600 span = 10
Multiplier Function = Tanh_Temperature x_ref = 600 span = 10 start = 100␣
↪end = 5
```

Thermodynamic Equilibrium

Note: `Thermodynamic_Equilibrium` is not supported with CHEMEQ

The `thermodynamic_equilibrium` multiplier function can be used to calculate the inverse equilibrium constant $\frac{1}{K_c}$ using thermodynamic properties. When multiplied by the forward rate function, this will provide a reverse rate function. The equation for K_c satisfies the Van't Hoff equation and uses Gibbs free energy.

$$f_m(t, T, P, area, Y) = \exp\left(-\sum_{i=1}^{n_s} \frac{\nu_i}{RT}(TS_i^o - H_i^o)\right)\left(\frac{RT}{P_{atm}}\right)^{\sum_{i=1}^{n_s}\nu_i}$$

In this equation, P_{atm} is the atmospheric pressure, R is the universal gas constant, n_s is the number of species, ν_i is the stoichiometric coefficient of the i^{th} species, and S_i^o and H_i^o are the standard state molar entropy and molar enthalpy of the i^{th} species respectively.

The two optional inputs for this are atmospheric pressure, P_{atm}, and the universal gas constant, R. The default values for these are 101325 Pa and 8.314 J/mol-K respectively. Heat capacity

evaluators must be provided with the species properties in order to use this function (the enthalpy and entropy evaluators will be derived from the provided heat capacity evaluators). Example inputs are shown below.

```
Multiplier Function = Thermodynamic_Equilibrium
Multiplier Function = Thermodynamic_Equilibrium  Patm = 101325 R = 8.314
```

Simplified Equilibrium

Note: `Simplified_Equilibrium` is not supported with CHEMEQ

The `simplified_equilibrium` multiplier function can be used to calculate the inverse equilibrium constant $\frac{1}{K_c}$ using a known value of the equilibrium constant (in terms of partial pressures) at a specific temperature and the heat of reaction. When multiplied by the forward rate function, this will provide a reverse rate function. The equation for K_c is the integrated Van't Hoff equation.

$$f_m(t,T,P,area,Y) = \left(\frac{1}{K_{eq}}\right) \exp\left(\frac{\Delta H^o}{R}\left(\frac{1}{T} - \frac{1}{T_{eq}}\right)\right)\left(\frac{RT}{P_{atm}}\right)^{\sum_{i=1}^{n_s} \nu_i}$$

In this equation, P_{atm} is the atmospheric pressure, R is the universal gas constant, n_s is the number of species, ν_i is the stoichiometric coefficient of the i^{th} species, K_{eq} is the known equilibrium constant (in terms of partial pressures) at temperature T_{eq}, and ΔH^o is the heat of reaction.

The required input for this is T_{eq}. The optional inputs include K_{eq}, P_{atm}, and R defaulted to 1.0, 101325 Pa, and 8.314 J/mol-K respectively. Either the heat of reaction or the enthalpies of formation must be provided with the reactions in the input file to use this equation accurately. Example inputs are shown below.

```
Multiplier Function = Simplified_Equilibrium  Teq=800
Multiplier Function = Simplified_Equilibrium  Teq=800 Keq=0.6
Multiplier Function = Simplified_Equilibrium  Teq=800 Keq=0.6 Patm=101325␣
↪R=8.314
```

General

The general multiplier function uses a function string supplied in the input file to evaluate the rate generally. Allowable variables in this string include `time` (time), `T` (temperature), `P` (pressure), area, and any valid species name from the current model. The value used for a species is in whatever unit system the general chemistry model is using. These functions can use all the functions available in *string function expressions*, but cannot access arbitrary expression values.

An example input for this option is

```
Multiplier Function = General  Function = 0.5*(1+tanh((T-350)^2))*max(T,
↪300)
```

Rate Functions

Unity

The rate function is the only required function, so it has no default like the others. However, you can still select Unity for it if desired, using

```
Rate Function = Unity
```

$$f_r(t, T, P, area, Y) = 1$$

Arrhenius

The Arrhenius rate function is the most common, and uses

$$f_r(t, T, P, area, Y) = AT^{\beta} \exp\left(\frac{-E_a}{RT}\right)$$

The pre-exponential constant, A, is the only required input. Optional inputs include E_a, β, and R. If omitted, the defaults for these are 0 for E_a and β and 8.314 for R. Some example lines to select this rate function is

```
Rate Function = Arrhenius  A = 100 Ea = 20000
Rate Function = Arrhenius  A = 100 Ea = 350 R = 1
Rate Function = Arrhenius  A = 100 beta = 2 Ea = 350 R = 1
```

Distributed Arrhenius

The Distributed Arrhenius rate function uses a cumulative normal distribution to vary the activation energy with respect to a selected progress species.

$$f_r(t, T, P, area, Y) = AT^{\beta} \exp\left(\frac{-(E_a + \sigma\zeta)}{RT}\right)$$

$$\zeta = CDF^{-1}(1 - Y_j/Y_0)$$

where CDF^{-1} is the clipped inverse cumulative normal distribution function, Y_j is the concentration of the selected progress species, Y_0 is the specified reference concentration, and σ is the activation energy variation. In addition to these inputs, the same input rules as for the Arrhenius reaction apply (only A is required of those again). The name specified for the progress species must be a valid species name.

Some example inputs for this rate type are (note that these are shown on multiple lines to fit in the manual, but must be on the same line in your input file):

```
Rate Function = Distributed_Arrhenius              \$
    A = 100 Ea = 20000                             \$
    sigma = 1000 Y0 = 1.2 ProgressSpecies = N2

Rate Function = Distributed_Arrhenius             \$
    A = 100 beta = 2 Ea = 350 R = 1               \$
    sigma = 100 Y0 = 1.2 ProgressSpecies = N2
```

General

The general multiplier function uses a function string supplied in the input file to evaluate the rate generally. Allowable variables in this string include `time` (time), `T` (temperature), `P` (pressure), area, and any valid species name from the current model. The value used for a species is in whatever unit system the general chemistry model is using. These functions can use all the functions available in *string function expressions*, but cannot access arbitrary expression values.

An example input for this option is:

```
Rate Function = General Function = 2.1*exp(-350/T)*N2^3.2*(1-CO2)*max(CO2,
↪0.1)
Rate Function = General Function = (T>300 ? 5.0 : 0.01)*CO2*exp(-350/T)
```

Pressure Functions

Unity

The default pressure function is unity,

$$f_p(P,T) = 1$$

and can be used by either omitting the `Pressure Function` line or using

```
Pressure Function = Unity
```

Exponential

The exponential pressure function is evaluated as

$$f_p(P,T) = \left(\frac{P}{P_{ref}}\right)^n,$$

where P_{ref} must be positive. An example input for this option is

```
Pressure Function = Exponential  P_ref = 1 n = 1.5
```

Concentration Functions

Unity

The default concentration function is unity,

$$f_c(Y) = 1$$

and can be used by either omitting the `Concentration Function` line or using

```
Concentration Function = Unity
```

Standard

The standard concentration function is evaluated as

$$f_c(Y) = \prod_{i=1}^{N_s} Y_i^{\mu_i}$$

The values of μ can be specified manually using a list of entries for μ. This is done by entering a list of name-value pairs for each entry with a non-zero μ value. Alternately, you can provide a list of μ values for all species.

If your reaction rate only depends on the concentration of the reactants, and the value of μ is the same as the stoichiometric coefficient, then you can have the values set automatically from the reaction stoichiometry by using the keyword `Automatic`.

For example, if your reaction is `2A + B -> C` the automatic option would do the same thing as specifying `mu = A 2 B 1`. If you have repeated species as both reactants and products, only the reaction portion of the coefficient is considered. For example, `A + B -> 2A + C` would result in automatic exponents of `mu = A 1 B 1`.

```
Concentration Function = Standard  mu = A 2 B 1
Concentration Function = Standard  mu = 2 1 0
Concentration Function = Standard  mu = Automatic
```

Prout Tompkins

The Prout Tompkins concentration function can only be used with a reaction that has a single reactant and a single product, and is evaluated as

$$f_c(Y) = \alpha^n \left(1 - \left(1 - 10^{-p}\right) \alpha\right)^m$$

where α is the concentration of the reactant. An example input for this model is

```
Concentration Function = Prout_Tompkins  n = 1.2 m = 2.5 p = 9
```

Evaporation

Note: `Evaporation` is not supported with CHEMEQ

The evaporation concentration function is evaluated as

$$f_c(Y) = A\sqrt{\frac{2}{\pi MRT}}\frac{1}{2 - C_c}\left(C_e p_v(T) x_{liq} - C_c p x_{vap}\right)$$

$$p_v(T) = p_0 \exp\left(\frac{L}{R}\left(\frac{1}{T_b} - \frac{1}{T}\right)\right)$$

where C_c and C_e are the condensation and evaporation coefficients, L is the molar enthalpy of vaporization, T_b is the boiling temperature at p_0, p is the gas phase pressure, A is the surface area,

x_{liq} is the mole fraction of the liquid that is evaporating, and x_{vap} is the gas phase mole fraction of the evaporating species. If L is provided in mass units, R must be converted appropriately so their units match. If you provide only C_c or only C_e, then the coefficients are assumed to be the same.

For reactions that do not include evaporation, species that are defined in the material but not involved in any reactions may be omitted from the chemistry definition block. When using evaporation reactions, this is not allowed since it will alter the value of the mole fractions in the equations above. You must also use a species variable name of `species` when using evaporation since this requires molar concentrations to get mole fractions.

The only rate, pressure, and multiplier functions allowed with an Evaporation concentration function are the Unity functions.

```
Concentration Function = Evaporation  R=8.314  M=18e-3  Cc=0.01  Tb=373 ␣
↪L=4e4
```

General Evaporation

Note: `General Evaporation` is not supported with CHEMEQ

This is a more general case of the evaporation function specified above. With general evaporation, different forms of the saturation pressure may be specified through a fugacity functor. The general evaporation function is evaluated as

$$f_c(Y) = A\sqrt{\frac{2}{\pi MRT}}\frac{1}{2-C_c}\left(C_e f_l - C_c f_v\right)$$

where f_l and f_v are the fugacity of the liquid and vapor respectively. Choices for the liquid fugacity include:

- `HENRY_CONSTANT`: returns $x_{liq}H : math :$ for a specified Henry's law constant H.
- `ANTOINE`: returns $x_{liq}10^{\left(A-\frac{B}{C+T}\right)}$ for specified Antoine coefficients A, B, and C.
- `VANT_HOFF_BOILING`: returns $x_{liq}P_{atm}\exp\left(\frac{L}{R}\left(\frac{1}{T_b}-\frac{1}{T}\right)\right)$ for required parameters of molar enthalpy of vaporization L and boiling temperature T_b at P_{atm}; and optional parameters of atmospheric pressure P_{atm} and universal gas constant R with default values of 101325 Pa and 8.314 J/mol-K respectively.

Choices for the vapor fugacity include:

- `IDEAL_GAS`: returns $x_{vap}P$ (no parameters to specify).

In the equation for $f_c(Y)$, inputs R, M, C_c, and C_e work the same as they would for the Evaporation case above. The rules defined above for using Evaporation also apply to `General_Evaporation`.

For an example reaction `Liq -> Gas`, general evaporation may be specified as

```
Concentration Function = General_Evaporation  R=8.314  M=18e-3  Cc=0.01
Begin Fugacity For Liq
    Type = VANT_HOFF_BOILING
    Parameter Patm = 101325
    Parameter R = 8.314
    Parameter Tb = 373
    Parameter L = 4e4
End
Begin Fugacity For Gas
    Type = IDEAL_GAS
End
```

which would give the same result as the Evaporation example given above. Other options for the liquid fugacity might look like

```
Begin Fugacity For Liq
    Type = ANTOINE
    Parameter A = 8
    Parameter B = 1000
    Parameter C = 200
End
```

```
Begin Fugacity For Liq
    Type = HENRY_CONSTANT
    Parameter H = 1
End
```

Condensation

Note: `Condensation` is not supported with CHEMEQ

The condensation concentration function is evaluated as

$$f_c(Y) = \frac{3DM(f_v - f_l)}{r^2 RT}$$

where f_l and f_v are the fugacity of the liquid and vapor respectively. Choices for the fugacity are given in the General Evaporation description above.

In the equation for $f_c(Y)$, D is the diffusion coefficient, M is the molar mass of the condensing species, r is the droplet radius, R is the universal gas constant, and T is temperature. The required parameter for condensation is D, while the optional parameter of R has the default value 8.314 J/mol-K.

Similar rules apply for using Condensation as using `General_Evaporation` defined above. However, you must use a mass basis for Y when using condensation and provide the molecular weights of all species present in order to calculate the mole fractions in the fugacity functions.

For an example reaction `Gas -> Liq`, condensation may be specified as

```
Concentration Function = Condensation  R=8.314  D=1e-4
Begin Fugacity For Gas
    Type = IDEAL_GAS
End
Begin Fugacity For Liq
    Type = VANT_HOFF_BOILING
    Parameter Patm = 101325
    Parameter R = 8.314
    Parameter Tb = 373
    Parameter L = 4e4
End
```

Heat Release

Reaction heat release can be specified by setting per-species formation enthalpies or by supplying a heat of reaction in each reaction block, using the `Heat of Reaction = X` command. The units of the provided value should be energy per mass (e.g. J/kg) when using mass-based units and energy per mol when using molar units.

Note: The sign convention for heat of reaction in the chemistry blocks is that endothermic reactions have a positive heat of reaction and exothermic reactions have a negative heat of reaction.

When supplying heat of reaction values, formation enthalpies will be calculated automatically. If you supply a set of heats of reaction for which formation enthalpies cannot be calculated you will get an error. For example, in the following scenario there are no formation enthalpies that could be constructed that satisfy the given heats of reaction.

```
Begin Reaction R1
  Reaction is A -> B
```

(continues on next page)

(continued from previous page)

```
  Heat of Reaction = 100
End

Begin Reaction R2
  Reaction is B -> A
  Heat of Reaction = 100
End
```

Similarly, using the same reaction stoichiometry twice with different heat release values will generate a similar error.

```
Begin Reaction R1
  Reaction is A -> B
  Heat of Reaction = 100
End

Begin Reaction R2
  Reaction is A -> B
  Heat of Reaction = 200
End
```

Sometimes one may want to use two identical reaction stoichiometries to represent two reaction pathways that occur at different temperatures. The best way to accomplish that is to create different product species.

```
Begin Reaction R1
  Reaction is A -> Blt
  Heat of Reaction = 100
End

Begin Reaction R2
  Reaction is A -> Bht
  Heat of Reaction = 200
End
```

Alternately, instead of specifying heats of reaction per reaction you can specify enthalpies of formation per species. You should not provide both heats of reaction and heats of formation however. Enthalpies of formation are supplied per species outside the reaction blocks, using a single line for CHEMEQ

```
species names are A B C D
Enthalpies of Formation = 0 3.5564e8 -7.1126e8 -99.5786e8
```

and multiple lines for General Chemistry

```
species names = A B C D
Enthalpy of Formation for A = 0
Enthalpy of Formation for B = 3.5564e8
Enthalpy of Formation for C = -7.1126e8
Enthalpy of Formation for D = -99.5786e8
```

Reaction Phases

Note: Reaction phase arguments are not supported with CHEMEQ

If the pressure phase argument is omitted but temperature phase is provided, the pressure phase defaults to the same phase as the temperature. By default, all of the reaction heating is applied in the temperature phase (e.g. solid phase). However, if you specify a value for the "Origin Phase Heat Fraction" between 0 and 1 you can divide that heat between phases. The fraction specified controls how much energy goes into the origin phase (a.k.a. temperature phase). The remaining energy is split evenly between the other phases that participate in the chemistry model.

Reversible Reactions

Note: Reversible reactions are not supported with CHEMEQ

If a reversible reaction is desired, the forward and reverse reaction must be specified in two separate reaction blocks. However, there is an option to say `Reaction is Reverse of [REACTION NAME]` giving the name of the forward reaction block. This will copy over the reaction string with the opposite signs for the stoichiometric coefficients, the rate function, the temperature and pressure phases, and the equal and opposite heat of reaction. If the forward reaction block contains a concentration function that is not Unity, this statement will also create a Standard concentration function for the reverse reaction with automatic coefficients. These default rate and concentration functions can be overwritten by specifying a new function inside the reverse reaction block. An example of how this option might be used for an equilibrium model is shown below.

```
Begin Reaction [FORWARD REACTION NAME]
    Reaction is A -> B
    Rate Function = [Required]
```

(continues on next page)

(continued from previous page)

```
    Concentration Function = [Optional]
    Heat of Reaction = [Value]
End
Begin Reaction [REVERSE REACTION NAME]
    Reaction is Reverse of [FORWARD REACTION NAME]
    Multiplier Function = [Thermodynamic_Equilibrium, Simplified_
↪Equilibrium] [args]
End
```

ODE Solvers

There are several ODE solvers available that can be selected from in Aria. The basic set of input arguments for an ODE solver are given below, where the specific solver arguments are described in the following subsections. Anywhere an ODE solver is requested in the input file, these additional arguments are also recognized.

```
ODE Solver = [SOLVER ARGS]
Absolute Tolerance = 1e-10
Relative Tolerance = 1e-5
Maximum Substeps = 100000
Minimum Step Size = 1e-15
```

Note: For GPU execution you should use one of the Runge-Kutta solvers since CVODE and LSODE are not supported on the GPU yet.

CVODE

The CVODE solver is the most flexible of the options available, and often the fastest. It is the only option that allows you to select the method (BDF or ADAMS), the order (max of 5 for BDF and 12 for ADAMS), and the iteration scheme (FUNCTIONAL or NEWTON). For stiff problems, `BDF 5 NEWTON` will be most robust, but is also the most costly. For non-stiff problems, a lower order Adams method, such as `ADAMS 12 FUNCTIONAL` may be better.

The default is `BDF 5 NEWTON` if you only write `ODE Solver = CVODE`. A few valid options are shown below.

```
ODE Solver = CVODE [METHOD] [ORDER] [ITERATION_SCHEME]
ODE Solver = CVODE ADAMS 12 FUNCTIONAL
```

(continues on next page)

(continued from previous page)

```
ODE Solver = CVODE BDF 3 NEWTON
ODE Solver = CVODE
```

LSODE

The LSODE solver can sometimes provide better performance than CVODE for large, stiff problems. It always uses the BDF 5 NEWTON scheme, so there are no extra options. The minimum time step in LSODE is set to machine precision and cannot be set by the user.

```
ODE Solver = LSODE
```

RKEH

The RKEH solver uses the Euler-Heun Method. It is a second-order error-adaptive Runge-Kutta scheme. It is typically slow for stiff systems, but may be advantageous for slowly changing non-stiff systems.

```
ODE Solver = RKEH
```

RK12

The RK12 solver uses the Fehlberg 1-2 Method. It is a second-order error-adaptive Runge-Kutta scheme. It is typically slow for stiff systems, but may be advantageous for slowly changing non-stiff systems.

```
ODE Solver = RK12
```

RKBS

The RKBS solver uses the Bogacki-Shampine Method. It is a third-order error-adaptive Runge-Kutta scheme. It is typically slow for stiff systems, but may be advantageous for slowly changing non-stiff systems,

```
ODE Solver = RKBS
```

RKF45

The RKF45 solver uses the Runge-Kutta-Fehlberg 4-5 Method. It is a fourth-order error-adaptive Runge-Kutta scheme. It is typically slow for stiff systems, but may be advantageous for slowly changing non-stiff systems,

```
ODE Solver = RKF45
```

CashKarp

The CashKarp solver uses the Cash-Karp Method. It is a fourth-order error-adaptive Runge-Kutta scheme. It is typically slow for stiff systems, but may be advantageous for slowly changing non-stiff systems.

```
ODE Solver = CashKarp
```

DOPRI5

The DOPRI5 uses the Dormand-Prince Method. It is a fourth-order error-adaptive Runge-Kutta scheme. It is typically slow for stiff systems, but may be advantageous for slowly changing non-stiff systems.

```
ODE Solver = DOPRI5
```

Pressurization Zones

Upon heating, materials like foams or energetic materials can decompose and produce gas that can accumulate in pore spaces and free volumes. A pressurization zone can be used to track the approximate pressure from this gas production using a simple molar balance and equation of state without the need to solve fluid dynamics equations. This pressure may simply be post-processed, or it may influence reaction kinetics for pressure-dependent reaction mechanisms.

Each pressurization zone requires a few pieces of information to be configured:

- A list of source blocks (blocks that produce gas)
- A list of pressurized blocks and bulk nodes (blocks with void space where gas can be contained)
- An equation of state (e.g. ideal gas or BKWS) to use to calculate pressure as a function of temperature, volume, and number of moles of gas
- A temperature averaging method (to determine the average temperature to use for the equation of state)

- Initial pressure (to determine the initial amount of gas in the zone)
- (optional) The volume and temperature of a non-meshed excess domain (e.g. a region where gas could accumulate that is not a part of the Aria model)

For each pressurization zone, a global variable for pressure, temperature, gas volume, and total moles will be created and shown in the log file. These can be output to heartbeat, included in the Exodus file, and used in user functions along with other global variables.

The complete set of commands for the pressurization model can be found in the *command summary section*.

Example

Consider the problem in the image below. Upon heating, the foam block (`block_4`) could produce gas that would flow into the void spaces in `block_4`, `block_3`, and the bulk node `air1`.

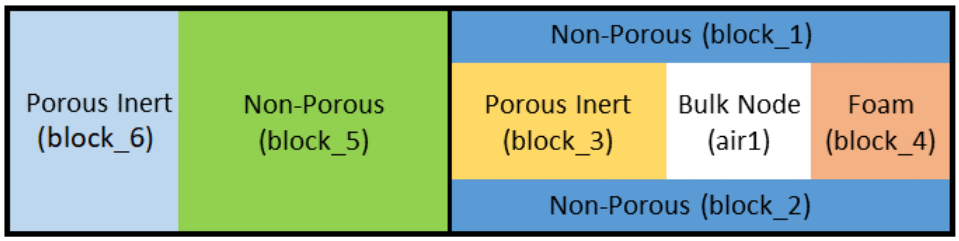


Fig. 4.28: Example domain for pressure zone setup.

To set up a pressurization zone to track the pressure of the gas in that domain, we would add the following pressurization model in the Aria Region, assuming the zone was initially at 14 psi:

```
Begin Pressurization Model Aft_Cavity
  Equation of State = Ideal_Gas
  Pressure Unit = psi
  Initial Pressure = 14
  Pressurization Source Blocks = block_4
  Pressurized Blocks = block_3 block_4
  Pressurized Bulk Nodes = air1
End
```

There are some required material properties for blocks involved in the pressurization zone:

- In order to calculate the gas volume, all the `Pressurized Blocks` must define a model for `volume fraction gas` in their material models.
- To calculate gas produced, all blocks in `Pressurization Source Blocks` must define a `SPECIES` expression for the molar concentration of produced gas (moles/volume) in their

material models. This could be defined from a chemistry model, level set state, or a custom user function.

- If a block has pressure-dependent reactions and needs the pressurization zone pressure, it should define `PRESSURE = PRESSURIZATION_MODEL` in the material model.

When using a CHEMEQ model, there are pre-defined `SPECIES` and `VOLUME FRACTION GAS` models that can be used to calculate these quantities based on the CHEMEQ model state. In the example below, the `SPECIES` is calculated based on the production of the species marked as "gas" in the CHEMEQ definition, and volume fraction of gas is calculated using the `solid_density` values and mass fractions of the solids taken from CHEMEQ.

```
Begin aria material foam # block_4
  Species Names = FOAM CHAR CO2

  Mass_Fraction of FOAM = From_ChemEq
  Mass_Fraction of CHAR = From_ChemEq

  Solid_Density of FOAM = Constant  rho_solid = 1200
  Solid_Density of CHAR = Constant  rho_solid = 1500

  Pressure = Pressurization_Model
  Volume Fraction Gas = From_Chemeq_Mass_Fractions
  Species = ChemEq_Gas

  # ...

  Begin Parameters for Chemeq Model foam_decomp
    species names are              FOAM      CHAR      CO2
    species phases are             Condensed Condensed Gas
    Species molecular weights are 1         1         120  # solid values␣
↪don't matter here

    # ...

    Concentration Units = Mass Fractions
  End
End

Begin aria material porous_inert # block_3
  volume fraction gas = 0.01

  # ...

End
```

Pressure Unit

The default pressure unit is Pascals (`Pa`). Other options are `psi` or `atm`. The pressure global variable will be in the units specified here, and will have the unit appended to the global variable name (e.g. `pZone1_pressure_psi`).

Temperature Averaging Method

This section describes the different averaging methods that can be selected to get an effective temperature (T_{eff}). In all cases, ϕ is the volume fraction of gas. This effective temperature is used in the equation of state to calculate the pressure in the pressurization zone.

Table 4.3: Temperature averaging methods

Method	Effective Temperature
`VOLUME`	$T_{eff} = \frac{\int \phi T dV}{\int \phi dV}$
`IDEAL_GAS`	$T_{eff} = \frac{\int \phi dV}{\int \phi / T dV}$
`RHO`	$T_{eff} = \frac{\int \phi \rho T dV}{\int \phi \rho dV}$
`RHOCP`	$T_{eff} = \frac{\int \phi \rho C_p T dV}{\int \phi \rho C_p dV}$

The `IDEAL_GAS` averaging is the most physically accurate method and is the recommended selection for all use-cases. It is the default if the averaging is not specified. It is derived by starting with the following relationships and solving for T_{eff}, assuming the pressure and compressibility (Z) are spatially uniform across the pressurized zone:

$$n_{tot} = \frac{PV_{tot}}{ZRT_{eff}} = \int \frac{P\phi}{ZRT} dV$$

$$V_{tot} = \int \phi dV$$

Equation of State

Given the volume, temperature, and number of moles of gas present in the pressurized zone, an appropriate equation of state is still required to calculate pressure. This is specified in the pressurization model block, and can be one of the options listed below. Any arguments required for the equation of state are given on the same line, after its name. All equations of state have the optional gas constant argument, R, which defaults to 8314 J/kmol-K if omitted. If your molar concentrations (`SPECIES`) are in units of moles, then you must supply R in J/mol-K (8.314).

Ideal Gas

The ideal gas equation of state requires no coefficients and evaluates the pressure as

$$p = \frac{nRT}{V}.$$

and is specified (with or without R) by

```
Equation of State = Ideal_Gas  R = 8.314
```

BKWS

The BKWS equation of state requires one parameter, the co-volume (V_{co}). The values of κ, β, and θ are constants (10.5e-3, 0.298, and 6620 respectively). The pressure is then evaluated using:

$$x = \frac{n\kappa V_{co}}{V\sqrt{T+\theta}}$$

$$Z = 1 + x\exp(\beta x)$$

$$p = \frac{ZnRT}{V}.$$

It is specified (with or without R) by

```
Equation of State = BKWS  covol = VALUE  R = 8.314
```

Van Der Waal

The VanDerWaal equation of state requires two parameters, a and b. The pressure is then evaluated using:

$$p = \frac{nRT}{V - bn} - a\frac{n^2}{V^2}$$

It is specified (with or without R) by

```
Equation of State = VanDerWaal  a = VALUE b = VALUE R = 8.314
```

Venting and Coupling

Aria has some basic capabilities to model venting of a pressurization zone to the environment, as well as to couple the pressure of multiple pressurization zones to one another to allow flow between zones. Venting may be modeled by setting `VENTING MODEL = VENTED` in the

pressurization model block. This also requires specifying a model for the venting flow rate as a function of the pressure in the zone. At present there are 2 supported models, one that accounts for choked flow and one that does not. Both models take the form:

$$\dot{m} = K\rho\sqrt{(P - P_{ambient})/\rho}$$

where K is an empirical factor, ρ is the density of the gas in the pressurization zone, P is the pressure in the pressurization zone, and $P_{ambient}$ is the pressure of the environment the gas is venting to. The choked flow model switches to the form:

$$\dot{m}_{choked} = K\rho\sqrt{P(1 - \frac{1}{P_{crit}})/\rho}$$

once $\frac{P}{P_{ambient}} > P_{crit}$ where P_{crit} is the user-specified critical pressure ratio. The value of K can be set using any of the generic material model forms supported by Aria (i.e. constant, polynomial, user function, etc.). A sample vented pressurization zone input deck block is presented below:

```
Begin Pressurization Model pZone1
  Initial Pressure = 13
  Initial Mixture Molecular Weight = 32
  Pressure Unit = psi

  Pressurization Source Blocks = block_1
  Pressurized Blocks = block_1

  Venting Model = Vented
  Venting Model Property Venting_Volumetric_Flow_Rate = K_FACTOR_WITH_
↪CHOKING \$
      critical_pressure_ratio=2.
  Venting Model Property ambient_pressure = constant value = 13
  Venting Model Property K_factor = constant value=1.e-2

  Equation of State = Ideal_Gas
End Pressurization Model pZone1
```

Multiple pressurization zones may be connected using the same model forms for the flow rate as are available for venting. This can be used to model multiple sealed volumes connected by a restricted flow path or feature that is initially sealed until it reaches a certain temperature or pressure for example. The coupling is turned on using the syntax:

```
Begin Bulk Node Coupling coupling
  Bulk Nodes = pZone1 pZone2
  Couple density model = k_factor_flow
  Couple species model = k_factor_flow
  additional parameter K_factor = user_function name=k_coupling X=time
End
```

where “pZone1” and “pZone2” are the names of 2 pressurization zones present in the model.

Flow Rate Limiter

When the K_factor value is large and the time step is large, the resulting high flow rate between zones may cause numerical convergence issues. To alleviate this, you can specify a flow rate limiter in your coupling specification, as

```
Begin Bulk Node Coupling coupling12
  Bulk Nodes = PZone1 PZone2
  Couple density model = k_factor_flow flow_rate_limiter=100
  Couple species model = k_factor_flow flow_rate_limiter=100
  additional parameter K_factor = Constant  value = 10
End
```

When using the limiter option, the code first calculates the amount of mass transfer between the two zones that would result in pressure equilibrium. This is divided by the current fluid time step to determine a baseline flow rate. The baseline flow rate is then multiplied by the limiter you specify to obtain the maximum allowable flow rate. In this example, the flow rate is limited to reaching equilibrium in 1 % of the current fluid time step. A larger limiter value results in less clipping of the flow rate (so less limiting). When this limiter is not provided, there is no flow rate limiting.

Coupling Algorithm

When the flow rates between coupled pressurization zones is large, it can cause numerical convergence issues in the overall nonlinear solution. When this is the case, you have the option of using a segregated solution approach for the coupling contributions to the pressurization zones. To use this option, you must first specify an ODE solver block in the Aria region, as shown in the example below. The syntax for this block is the same as for the chemistry ODE solver blocks, and is described in *the chemistry section*

```
Begin ODE Solver Parameters PZone
  ODE Solver = CVODE
  Absolute Tolerance = 1e-15
  Relative Tolerance = 1e-9
End ODE Solver Parameters
```

Next, you must select the segregated coupling algorithm in each pressurization zone involved in the coupling network and list which ODE solver to use.

```
Begin Pressurization Model pZone1
  Coupling Algorithm = Segregated
```

(continues on next page)

(continued from previous page)

```
  Solver Name = PZONE
End Pressurization Model pZone1
```

There are a few notes to be aware of when using this option:

- The coupled network will use the ODE solver specified in the first block it stores, so if you specify different solvers in each pressurization model some of them will be ignored.
- The segregated approach only works for CLOSED pressurization zones. If you were using a VENTED pressurization zone in your network to vent to atmosphere, you will need to add a new pressurization zone to represent atmosphere (with a large excess volume and no listed blocks) and vent to that instead.
- For very high flow rates, the flow rate limiter described previously may still be required for good convergence.

Boundary Coupling

A pressurization zone can also be coupled to a region of the domain where fluid transport is solved through a coupling boundary. The pressurization zone pressure will be applied as the boundary condition pressure for the fluid solve, and any mass flow through the boundary will be added or subtracted from the pressurization zone molar balance.

To use this option, the porous domain should use a bulknode_flux boundary condition with the name of the pressurization zone as its target.

```
BC bulknode_flux for mass_balance in gas_phase on surface_1 = bulk_node_
↪mass_open bulk_node = pZone1
```

Submodel

In the course of system analysis simulations, situations may arise in which the addition a certain feature may hinder the creation of a meshed discretization. Although the feature is needed, an alternative approach is to approximate its effect by including the feature as a submodel of the more comprehensive background model. Here the feature in question is meshed independent of the background discretization and is included in the simulation by constraining local submodel node DOF to the background model DOF nodes. Hence node DOF on the embedded element blocks are constrained to node DOF on the enclosing element blocks.

```
# Scope: Sierra > Procedure > Aria Region
Begin Submodel aria-embed
  embedded blocks = block_101
  enclosing blocks = block_1
```

(continues on next page)

(continued from previous page)

```
  ...
End
```

PID Controller

Aria provides a classical parallel PID (Proportional-Integral-Derivative) for a number of source and BC commands, commonly used to recreate similar controllers used in experiment. The controller aims to maintain a desired output level by minimizing the difference between a measured process variable and a desired setpoint. This implementation involves three parallel components:

Proportional (P)

- This part of the controller reacts to the current error, which is the difference between the setpoint and the process variable.
- The proportional gain K_p determines the magnitude of the response to the error. A higher K_p results in a larger output response, which can reduce the rise time but may increase overshoot and system instability.

Integral (I)

- This component addresses the accumulated error over time, effectively eliminating the steady-state error that can occur with a P-only controller.
- The integral gain K_i determines how aggressively the controller reacts to the accumulated error. By integrating the error over time, this component helps in achieving zero steady-state error, but too high a K_i can lead to overshoot and oscillations.
- Optional anti-windup is provided here to saturate the integrated error, minimizing overshoot when the error is large over a long period of time. This is enable by setting `enable_clamping=true` in the PID line command.

Derivative (D)

- The derivative part predicts future errors by calculating the rate of change of the process error.
- The derivative gain K_d influences the controller's response to the rate at which the error is changing. This component can improve system stability and reduce overshoot by "damping" the response, especially in systems with a fast-changing process variable.
- The derivative action acts on a filtered value of the error, avoiding spikes in command caused by a noisy process variable (e.g. those coming from sensor data). The time constant of this filter is controlled by specifying `Filter_Tau` or `Filter_Number` (where $\tau_f = 1/N_f$) in the PID line command.

In the parallel PID implementation used here, the outputs of these three components are summed to generate the control action to the process

$$u(t) = K_p e(t) + K_i \int_0^t e(\tau)d\tau + K_d \frac{\partial e_f(t)}{\partial t}$$

For example to apply a *PID controller heat flux* on a surface to recreate a heater used in experiment, one would use the following

```
Begin Definition for Function target_temp
    Type is Piecewise Linear
    Column Titles Time Temperature
    Begin Values
      0 300
      1 305
      ...
    End
End
...

postprocess average of expression temperature on heater_surface as avg_
↪heater_surf_temp
BC Flux for Energy on heater_surface = PID_Controlled \$
  P=1e-3 I=1e-5 D=0.0 \$
  Max_Output=1e6 \$
  Filter_Tau=60.0 \$
  Setpoint_Function_Name=target_temp \$
  Control_Variable=avg_heater_surf_temp
```

Note on Labview Controller

Unfortunately, a bug was uncovered in the implementation of the built in Labview PID controller that causes a deviation of the controller from classical PID theory when an integral and/or derivative term is included. To work around this bug, one must explicitly set a sampling dt in labview, use their built in conversion tool from Labview gains to parallel gains (also with the dt wired in), and still must then scale the gains by dt. Reference TF-3293 and/or Labview Request #: 02780637 for more context on this.

This workaround can be quite an unclear and burdensome process. Furthermore, it does not help for existing results generated with the broken controller. To help in this case, a helper script is provided to solve for the appropriate gains that best match the labview curves. The tool is available with the `sierra` module e.g.

```
module load sierra
labviewPID2aria --help
```

Note: This tool is provided as is. While we are not aware of any issues, we make no guarantees of correctness.

Consider the following CSV output from an experiment run in Labview controlling `temperature` to match `set_point` by changing `heater_cmd` over `time`

```
time, foo, temperature, heater_cmd, set_point, ...
0   , 1.0, 0          , 1         , 1        , ...
0.1 , 1.0, 0.1        , 1         , 1        , ...
...
```

To solve for the appropriate Aria gains, one needs only to feed in the initial guess generated by Labview and clarify the mapping of data e.g.

```
labviewPID2aria \
  --csv-data /path/to/data.csv \
  --time-key time \
  --response-key temperature \
  --cmd-key heater_cmd \
  --setpoint-key set_point \
  --const-P 20.0 \
  --solve-I 10.0 \
  --solve-D 1.0 \
  --clamping 100.0
```

Some notable optional capabilities include:

`--const-*`
: Provide a constant value for a given gain, do not solve for it.

`--solve-*`
: Provide an initial guess for a given gain, include it in the solve.

`--figure-out`
: Optionally write a figure demonstrating the resulting match. Since a *perfect* match cannot always be found, it is suggested to use this and confirm that the match is acceptable.

`--aprepro-out`
: Optionally write the results to an aprepro file to include in a run.

`--clamping`
: Enable clamping with specified command bounds.

See `labviewPID2aria --help` for a complete set of options. To handle this same data stored in an Excel spreadsheet, one needs only to change the data specification type to e.g.

```
labviewPID2aria --xlsx-data /path/to/data.xlsx:my_sheet_name ...
```

Note: The `xlsx-data` switch expects a `<path>:<sheet name>` tuple as argument. If the `:<sheet name>` is omitted, the first sheet in the file is selected instead.

4.5 User Customization

String Functions

String functions are one of the most flexible generic expressions supported in Aria for most material properties, initial conditions, sources, boundary conditions, and post-processors. String functions can be used to define scalar, vector, and tensor expressions as a function of position, time, and any expression that has been registered from the input deck. Currently the arguments for the string function must be scalars, although in the future vectors and tensors will be supported. If the function contains commas or spaces, it should be enclosed in quotes. Sensitivities for these string functions are calculated using finite-differences.

Valid inputs to a string function (not case-sensitive) are:

- `x` - X coordinate (includes deformation)
- `xref` - X coordinate (non-deformed)
- `y` - Y coordinate (includes deformation)
- `yref` - Y coordinate (non-deformed)
- `z` - Z coordinate (includes deformation)
- `zref` - Z coordinate (non-deformed)
- `t` - Current time
- `dt` - Current time step
- `t_old` - Time at the prior time step
- Any valid scalar expression name (e.g. `DENSITY`), old state of a valid scalar expression (e.g. `OLD_DENSITY`), or scalar component of a vector (e.g. `GRAD_TEMPERATURE_X`)

String functions that you define automatically provide their corresponding gradient and old time values, and time-dependent inputs are shifted automatically. For example, if you define

```
density = scalar_string_function f = "temperature + t"
```

Aria will automatically also define `OLD_DENSITY = OLD_TEMPERATURE + T_OLD`, so they can safely be used in terms that require a time derivative or gradient.

Scalar String Functions

Some examples of correct syntactic usage of these scalar string functions are listed below. For example, to define a spatially varying density you could use:

```
density = scalar_string_function f = "2*x"
```

Here, the material property `density` was assigned a value depending on the `x` coordinate. To define thermal conductivity as a function of temperature, you could use:

```
density = scalar_string_function f = "5*ln(temperature)"
```

Using *User Expressions* in combination with string functions lets you create more complicated property models. For example, to combine some tabulated values from a *User Function* called `DensityScaling` and an analytical function to get thermal conductivity you could do:

```
user expression = user_function user_tag = "rho_scaling" \$
    name = "DensityScaling" X = "Density"

user expression = scalar_string_function user_tag = "intrinsic_k" \$
    f = "3 + 4*ln(TEMPERATURE) - 3e-2*TEMPERATURE"

thermal conductivity = scalar_string_function \$
    f = "intrinsic_k * rho_scaling"
```

You can also use vector components and dot products with the string functions. For example, to use components of the temperature gradient you could do:

```
thermal conductivity = scalar_string_function f = "5 + GRAD_TEMPERATURE_X"
```

To take a dot product of two vector quantities and use it in another function:

```
user expression = dot_product user_tag = "GTdotGV" \$
    A = "Grad_Temperature" B = "Grad_Voltage"

thermal conductivity = scalar_string_function f = "5 + GTdotGV"
```

These string functions can also be used for boundary conditions and initial conditions.

```
BC dirichlet for temperature on surface_1 = scalar_string_function \$
    f = "density*sin(t)"
```

Here, a dirichlet boundary condition was applied on a surface for temperature as a function of the material property density (defined in the first example) and time. You can insert any expression that exists in your input deck as an argument for the function. If the expression is not found, the program will exit with an error about a missing expression.

String functions can also be used for initial conditions:

```
IC for temperature on block_1 = scalar_string_function f = "x*y*z"
```

Here, an initial condition for temperature is set also as a function of the spatial coordinates. This is for a three-dimensional case as the "z" component is included. If this line was used in a two-dimensional geometry, the program would exit because it could not find the "z" coordinate.

Vector String Functions

Vector string functions work in the same manner as their scalar counterpart, but must have functions specified for each vector component. For example, you could use a vector string function to apply a Dirichlet BC for velocity or any other vector quantity.

```
BC dirichlet for velocity on surface_1 = vector_string_function \$
    f_x = "1-x" f_y = "viscosity*sin(t)" f_z = "0.0"
```

Tensor String Functions

Tensor string functions work in the same manner as vectors, but must have functions specified for each tensor component.

```
Intrinsic Permeability = tensor_string_function \$
    f_xx = "1-x"                                \$
    f_yy = "1e-12*exp(t)"                       \$
    f_xy = "solid_phase_porosity*1e-12"         \$
    f_yx = "0"
```

Here we define the intrinsic permeability as a tensor string function for a two-dimensional geometry. For the `tensor_string_function`, the user must supply all components of the tensor. However, one can use the `tensor_string_function_symmetric`` command which enforces symmetry on the off-diagonal tensor component such that `f_yx = f_xy`.

```
Intrinsic Permeability = tensor_string_function_symmetric \$
    f_xx = "1-x"                                          \$
    f_yy = "1e-12*exp(t)"                                 \$
    f_xy = "solid_phase_porosity*1e-12"
```

Global Variables In String Functions

You can access global variables in an expression string function by adding the `*.global` suffix to the variable argument in the string function. For example, given a global variable named `my_gvar`, we will use it in the `density` property of a material:

```
density = scalar_string_function f = "temperature/my_gvar.global"
```

This will use the current value of `my_gvar` in this density model. Note, if this global is obtained through a postprocessor, it will use the global value of the previous timestep.

If you were using a global user expression to do this previously, you can replace that with the current approach.

String Functions for Contact, Bulk Node and Interface Boundary Conditions

There are several use cases where boundary conditions are being defined at an interface between two blocks: *contact enforcement*, *bulk node boundary conditions*, and *interface boundary conditions*. Those boundary conditions often need to be able to reference quantities from the block on either side of that interface. This is supported in string functions via `variable_name.location` syntax, for example:

```
BC bulknode_flux for energy on surface_1 = scalar_string_function bulk_
↪node="bulk1" f="10 * (temperature.block_1 - temperature.bulk1)"
```

would apply a convective condition between surface_1 and the bulk node bulk_1 on a mesh where surface_1 is on the exterior of block_1.

```
Enforcement for current = scalar_string_function f="10 * (voltage.current␣
↪- voltage.opposite)"
```

would apply a contact resistance condition between the two sides of the interface.

The following options are always valid location specifiers for a variable in any of the interface condition types:

- The name of one of the blocks on either side of the interface, either a user-specified name in the mesh or canonical name (i.e. block_1)
- A mesh group or assembly name that uniquely identifies the block (i.e. the mesh group cannot contain the blocks on both sides of the interface)

In addition for bulk node boundary conditions you can use:

- The name of the bulk node

And for contact enforcement you can use:

- “current” to refer to the side of the interface the enforcement condition is being applied to.
- “opposite” to refer to the side of the interface found via the contact search algorithm.

Note that generalized contact enforcement is symmetric so the string function condition will be applied once in each direction.

String Function Syntax Guide

String function evaluation is handled using the STK expression evaluator. In addition to the registered variables mentioned in the prior sections, there are a large number of standard functions and operations available in the evaluator. The following section shows a number of different operators with examples and descriptions.

Powers

Powers can be specified using the `pow` function or the standard `a^b` notation.

```
3^2
pow(3,2)
```

Min and Max Functions

There are min and max functions that take 2 to 4 comma-separated inputs.

```
min(1,2)
min(1,2,3,4)
max(1,2)
max(1,2,3,4)
```

Inline Variables

You can define variables inline in the function string, although doing this with aprepro in your input file is often more straightforward.

```
a=1;b=2;a+b+4
```

Ramps and Pulses

There are a number of pre-defined ramp and step functions.

```
cycloidal_ramp(t,ts,te)
haversine_pulse(t,ts,te)
cosine_ramp(t)
cosine_ramp(t,te)
cosine_ramp(t,ts,te)
linear_ramp(t,ts,te)
sign(x)
unit_step(t,ts,te)
point2d(x,y,r,w)
point3d(x,y,z,r,w)
```

Cosine Ramp

The `cosine_ramp` function provides a smooth ramp from 0 to 1. It can be used with 1, 2, or 3 inputs. The form with one input uses a start and end time of 0 and 1. The form with two inputs specifies the end time, but uses a start time of 0. Note that `cos_ramp` is also a valid function, and is equivalent to `cosine_ramp`.

$$\text{cosine_ramp}(t, t_s, t_e) = \begin{cases} 0, & t \le t_s \\ \frac{1}{2}\left(1 - \cos\left(\frac{\pi(t-t_s)}{t_e-t_s}\right)\right), & t_s < t < t_e \\ 1, & t \ge t_e \end{cases}$$

Linear Ramp

The `linear_ramp` function provides a linear ramp from 0 to 1.

$$\text{linear_ramp}(t, t_s, t_e) = \begin{cases} 0, & t \le t_s \\ \frac{t-t_s}{t_e-t_s}, & t_s < t < t_e \\ 1, & t \ge t_e \end{cases}$$

Cycloidal Ramp

The `cycloidal_ramp` function is another smooth ramp function from 0 to 1. Unlike the `cosine_ramp` function, it requires all three arguments.

$$\text{cycloidal_ramp}(t, t_s, t_e) = \begin{cases} 0, & t \le t_s \\ \frac{t-t_s}{t_e-t_s} - \frac{1}{2\pi}\sin\left(\frac{2\pi(t-t_s}{t_e-t_s}\right), & t_s < t < t_e \\ 1, & t \ge t_e \end{cases}$$

Haversine Pulse

The `haversine_pulse` function is a smooth sinusoidal finite width pulse, defined by

$$\text{haversine_pulse}(t, t_s, t_e) = \begin{cases} 0, & t \le t_s \\ \sin\left(\frac{\pi(t-t_s)}{t_e-t_s}\right)^2, & t_s < t < t_e \\ 0, & t \ge t_e \end{cases}$$

Sign

The `sign` function returns the sign of its single argument

$$\text{sign}(x) = \begin{cases} 1, & x \ge 0 \\ -1, & x < 0 \end{cases}$$

Unit Step

The `unit_step` function is a sharp square step function

$$\text{unit_step}(t, t_s, t_e) = \begin{cases} 0, & t \le t_s \\ 1, & t_s < t < t_e \\ 0, & t \ge t_e \end{cases}$$

2D Point

The `point2d` function provides a 2D point mask for applying a function in a certain spatial window. It is equivalent to a longer call to `cosine_ramp` where

$$\text{point2d}(x, y, r, w) = 1 - \text{cosine_ramp}(\sqrt{x^2 + y^2}, r - w/2, r + w/2)$$

3D Point

The `point3d` function provides a 3D point mask for applying a function in a certain spatial window. It is equivalent to a longer call to `cosine_ramp` where

$$\text{point3d}(x, y, z, r, w) = 1 - \text{cosine_ramp}(\sqrt{x^2 + y^2 + z^2}, r - w/2, r + w/2)$$

Basic Math

There are standard mathematical functions. Most of these require no explanation, although it is worth noting that `log` is the natural log (equivalent to `ln` not `log10`).

```
exp(x)
ln(x)
log(x)
log10(x)
pow(a,b)
sqrt(x)
erfc(x)
erf(x)
acos(x)
asin(x)
asinh(x)
atan(x)
atan2(y,x)
atanh(x)
cos(x)
cosh(x)
acosh(x)
sin(x)
sinh(x)
tan(x)
tanh(x)
```

Rounding Functions

There are a variety of rounding and numerical manipulation routines available.

```
ceil(x)
floor(x)
abs(x)
mod(x,y)
```

(continues on next page)

(continued from previous page)

```
ipart(x)
fpart(x)
```

You can round up with `ceil` and round down with `floor`. You can separate a number into its integer part (`ipart`) and floating point part (`fpart`). You can compute the modulus with `mod` and get the absolute value with `abs`.

Polar Coordinate and Angle Helpers

When working with polar and rectangular coordinates there are helper functions for converting coordinate systems as well as for converting degrees to radians and back.

```
poltorectx(r,theta)
poltorecty(r,theta)
deg(r)
rad(d)
recttopola(x,y)
recttopolr(x,y)
```

The conversion functions are

$$\text{poltorectx}(r, \theta) = r \cos(\theta)$$

$$\text{poltorecty}(r, \theta) = r \sin(\theta)$$

$$\text{recttopola}(x, y) = \text{atan2}(y, x)$$

$$\text{recttopolr}(x, y) = \sqrt{x^2 + y^2}$$

The angle returned by `recttopola` is adjusted to be between 0 and 2π.

Distributions and Random Sampling

There are several functions available for generating pseudo-random output with different distributions. When using random output in a string function, be careful using it in places where it can affect nonlinear convergence.

For example, setting an initial condition using a `random` distribution is acceptable since it is only evaluated once. However, using that to define a material property (for example, thermal conductivity) or boundary condition would result in that property varying not just in time and space, but also from one nonlinear iteration to the next, which could keep a Newton solver from converging. The `ts_random` and `ts_normal` functions are designed to help with this problem.

```
weibull_pdf(x,shape,scale)
normal_pdf(x,mu,sigma)
gamma_pdf(x,shape,scale)
log_uniform_pdf(x,xmin,xmax)
exponential_pdf(x,beta)
random()
random(seed)
time()
ts_random(t,x,y,z)
ts_normal(t,x,y,z,mu,sigma,minR,maxR)
```

Weibull Distribution

The `weibull_pdf` function returns a Weibull distribution, where λ is the scale parameter and k is the shape parameter.

$$\text{weibull_pdf}(x, k, \lambda) = \begin{cases} \frac{k}{\lambda}\left(\frac{x}{\lambda}\right)^{k-1} e^{-(x/\lambda)^k}, & x \geq 0 \\ 0, & x < 0 \end{cases}$$

Normal Distribution

The `normal_pdf` function returns a normal distribution, where μ is the mean and σ is the standard deviation.

$$\text{normal_pdf}(x, \mu, \sigma) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right)$$

Gamma Distribution

The `gamma_pdf` function returns a gamma distribution, where θ is the scale parameter and k is the shape parameter.

$$\text{gamma_pdf}(x, k, \theta) = \frac{1}{\Gamma(k)\theta^k} x^{k-1} e^{-x/\theta}$$

Log Uniform Distribution

The `log_uniform_pdf` function returns a log-uniform distribution

$$\text{log_uniform_pdf}(x, x_0, x_1) = \frac{1}{(\ln(x_1) - \ln(x_0))x}$$

Exponential Distribution

The `exponential_pdf` function returns a one-parameter exponential distribution

$$\text{exponential_pdf}(x, \beta) = \frac{1}{\beta} e^{-x/\beta}$$

Random Values

There are several functions for generating pseudo-random numbers. These can be used as inputs to any of the distributions, or as standalone values to get a uniform distribution.

`random()` calls a platform-independent fast pseudo-random number algorithm. It is seeded with a static constant, so the first call to it will always produce the same value. The value returned will be between 0 and 1.

`random(seed)` calls the same algorithm as `random()` but sets the seed first then returns a number between 0 and 1. The seed in this case can be any floating point number.

`time()` returns the current time integer. This can used to provide a random number seed to make the distributions non-deterministic from run to run.

`ts_random(t,x,y,z)` returns a uniformly distributed random number from 0 to 1 that is unique in time and space. This means that repeated evaluations of this function at the same point in time and space will produce the same output. This is useful for applying randomness on boundary conditions, for example, where the value should not change during nonlinear iterations at a given time.

`ts_normal(t,x,y,z,mu,sigma,minR,maxR)` returns a clipped normally distributed random number with a mean of `mu` and standard deviation of `sigma`. The output is clipped to be between `minR` and `maxR` and, like `ts_random`, the output is deterministic in time and space.

Boolean and Ternary Logic

In addition to all these functions, you can make more complicated functions using ternary statements and boolean logic.

You can use ternary operators for simple conditional assignment. The basic structure of a ternary operator is `boolean ? value_if_true : value_if_false` For example, the following if-else code

```
if x > 2.2:
  return 1.2*y
else
  return 0.1*y
```

can be converted to a ternary operator that returns the same thing

```
(x > 2.2 ? 1.2 : 0.1) * y
```

You can also use boolean operations in the string functions. By default, booleans that evaluate to true are treated as 1 and booleans that are false are 0. The prior example can be expressed using boolean statements as

```
(x>2.2)*1.2*y + (x<=2.2)*0.1*y
((x>2.2)*1.1 + 0.1)*y
```

These operations can be combined (multiplication of boolean statements is equivalent to a logical and). To replicate the `unit_step` function described earlier you can use either ternary or boolean operators. The following three functions all produce the same result - a step of height 5 between 1 and 2.

```
5*unit_step(x,1,2)
((x>1 && x<2) ? 5 : 0)
5*(x>1)*(x<2)
```

Warning: Note that chained inequality syntax is not allowed:

```
1 < x < 2       // parse error
(1<x) && (x<2)  // correct
```

Language Idiosyncrasies

There are a few places where the language rules used in the STK evaluator differ from those used in common tools like Python or Excel. These are noted in the sections below.

Modulus

For negative numbers, there is not a consensus among programming languages about the appropriate behavior for the modulus operator. Consider the statement `-30 % 360` - should this return 330 or −30? In Python and Excel this would return 330, but in C and the string functions this will return −30.

To produce a Python-like modulus in a string function for `x % 360` you would need to use `x - 360*floor(x/360)` instead.

Powers of Negative Constants

An expression like `-1^2` has some ambiguity - it could be treated as `(-1)^2` or `-(1^2)`. Again, here the string function differs from Python and Excel - which treat the - as a leading prefix and not as part of the numeric constant (so they return −1). The STK evaluator treats the - as part of the numeric constant, so it evaluates it as `(-1)^2` and returns 1.

However, the STK evaluator treats the - as a subtraction operator when it joins two terms, so in an expression like `0 - 1^2` it would return −1.

Warning: Always use parentheses in expressions like these to ensure consistent and un-ambiguous behavior across languages.

User Expressions

By default, properties defined in Aria provide an expression with the same name. For example, `density = constant value = 1` will provide an expression for `DENSITY`. Aria defines many property expressions that can be defined in this way, but sometimes for post-processing or complex intermediate quantities a user may want to provide their own expression with an arbitrary name. The user expression provides this capability.

One common usage is to do unit conversion - for example if you want to use temperature in Celsius as an input to a table function, you could do

```
user expression = scalar_string_function user_tag = "Tcelsius" \$
    f = "temperature - 273.15"

density = user_function name = my_rho_Tc_table X = Tcelsius
```

Here Tcelsius could later be used with a postprocessor to populate a nodal field containing the temperature in Celsius (assuming the model is defined in SI units).

```
postprocess Tcelsius on block_1
```

The `user_tag` argument defines the name of the provided expression. There are some rules about what this name can be. In general, you should not use a reserved suffix like `_X` or a species name (e.g. `my_foo_x` or `my_foo_CO2`) or a reserved prefix like `GRAD_` or `OLD_`.

Warning: If you provide a user tag that conflicts with an existing expression (e.g. `user_tag = "density"`) you will most likely get an error about a duplicate provided expression, however in rare cases your contribution may be silently added to the other expression.

Universal Expressions

While *User Expressions* can be used in Aria materials, they are often better suited to be applied globally. For example, if you provide a user expression to convert temperature from Kelvin to Celsius it would be repetitive to have to define that in every material. The universal expression block (at the Sierra scope in the input file) allows you to define such expressions that will be valid on all blocks, regardless of their material type.

```
Begin universal aria expressions
  User Expression = scalar_string_function user_tag = "Tcelsius" f =
↪"temperature - 273.15"
  User Expression = scalar_string_function user_tag = "radius_cm" f =
↪"sqrt(x^2 + y^2)*0.01"
  Axial Coordinates = Vector_Aligned axis = 1 0 0 origin = -0.7 0 0 ref =␣
↪0 1 0
End
```

Expressions defined at this scope are available to use in all material properties, boundary conditions, sources, initial conditions, or post-processors. The only types of Expression that are allowed in the universal expression block are `user expression` and `axial coordinates`. Outside of string functions, coordinates *x, y, z* referenced in models should use *coordinates_x, coordinates_y, coordinates_z* instead.

User Functions

User functions are 1D tables specified directly in the input file (at the Sierra scope) that can be used wherever generic expressions are available in Aria (e.g. material properties, boundary conditions, initial conditions, sources). These functions can be either piecewise linear (most common) or piecewise constant (step functions).

A user function is defined at the Sierra-level in the input file. Examples of piecewise linear and piecewise constant functions are shown below. Evaluations outside the table bounds are clipped to the table boundary, not extrapolated.

```
Begin Sierra Aria

  # ...

  Begin Definition for Function spec_heat_ss304
    Type is Piecewise Linear
    Begin Values
      293.0  500
      433.0  800
    End Values
  End
```

```
Begin Sierra Aria

  # ...

  Begin Definition for Function heat_flux
    Type is Piecewise Constant
    Begin Values
      0.0  0
      1.0  500
    End Values
  End
```

These functions can be used for material properties using the `user_function` model:

```
specific heat = user_function name = spec_heat_ss304 X = temperature
```

or in boundary conditions or other places with a similar syntax:

```
BC Flux for energy on surface_1 = user_function name = heat_flux X = time
```

The full set of syntax options for user functions can be found in the *User Functions* command reference section, which includes additional commands for scaling and offset options for both data columns and several less commonly used options.

Multi-dimensional Tables

Multi-dimensional structured tables can be used to define multivariate scalar functions in a tabulated form and are useful for when the exact mathematical relationship between inputs and the output is not readily available (e.g., experimental data). The structured tables presented here are analogous to 1-dimensional tabulated scalar user functions (e.g. $f(x)$) but instead support up to three independent variables (e.g., $f(x, y, z)$). Functions that output vector or tensor quantities are not supported.

The tabulated function is interpolated using multi-dimensional Lagrange interpolants provided by the tabular properties library. By default, a tensor product of linear 1-dimensional interpolants is used resulting in linear, bi-linear, and tri-linear interpolations for one, two and three dimensions respectively. Using the interpolants, function and derivative evaluations are available at any point inside the table domain. For evaluation points outside the table domain, the independent variables will be clipped to the corresponding maximum or minimum table extent and the function will be evaluated at the clipped point. Similarly, derivatives will be clipped to zero. As with user functions, no warnings are printed to the log file when this happens.

As shown in the examples section, multi-dimensional table expressions are accessible through the model `ND_Table` and can be used anywhere other generic expressions are provided (e.g., initial conditions, boundary conditions, sources, materials, user expressions). Before covering concrete examples, we first detail the expected file format (HDF5) and the structure of the multi-dimensional tables.

HDF5 Table Format

An HDF5 file can contain multiple named "datasets", each containing an array of data of arbitrary size and shape. An HDF5 file used for ND Tables in Aria must be structured using the following rules:

- The independent variables and dependent variable must be stored as HDF5 datasets under the root HDF5 group `/`.
- The independent variables should be stored in 1-D arrays in datasets named "i0", "i1", and "i2", of lengths M, N, and L, respectively.
- The independent variable arrays must be strictly increasing, but do not need to be uniformly spaced.
- The independent variable arrays must have a length of at least 2. Dimensions of length 1 should not be included in the table.
- There must be at least 1, and not more than 3, independent variable arrays.
- The dependent variable must be stored in an $M \times N \times L$ dataset named "function".

For example, for tabulating a scalar function of two independent variables, three datasets must be provided: a 1-dimensional array named "i0" of length M, a 1-dimensional array named "i1" of

length N, and a 2-dimensional array of size $M \times N$ named "function" evaluated at the structured grid formed by cartesian product of the two 1-dimensional arrays "i0" and "i1".

HDF5 Table Generation

Below we illustrate how to generate a proper HDF5 table using python with the h5py and numpy modules. These python modules are available in the python distribution locally provided by anaconda3 (i.e., `module load apps/anaconda3`). Along side the python code snippets are the corresponding file contents of each table obtained using h5dump. It is recommended to inspect the table that is written to ensure proper data layouts were used. Note the dimensionality and datatype (double) are determined by the numpy arrays.

Warning: HDF5 tables generated in C++ or Python use a row-major data arrangement (which is what Aria expects). If you generate your table using MATLAB it outputs a column-major data arrangement and the table will be transposed.

1D Example

Generating a 1D hdf5 table in python:

```
import h5py
import numpy as np

f = lambda i0 : -0.1 * (i0 - 300) + 100.0
temp = np.linspace(300, 600, 4, dtype='d')
rho = f(temp)

with h5py.File("1d_example.h5", 'w') as h5File:
  h5File.create_dataset("function", data=rho)
  h5File.create_dataset("i0", data=temp)
```

Corresponding h5dump output for the 1D example table:

```
HDF5 "1d_example.h5" {
GROUP "/" {
  DATASET "function" {
      DATATYPE  H5T_IEEE_F64LE
      DATASPACE  SIMPLE { ( 4 ) / ( 4 ) }
      DATA {
      (0): 100, 90, 80, 70
      }
```

(continues on next page)

(continued from previous page)

```
  }
  DATASET "i0" {
      DATATYPE  H5T_IEEE_F64LE
      DATASPACE  SIMPLE { ( 4 ) / ( 4 ) }
      DATA {
      (0): 300, 400, 500, 600
      }
  }
}
}
```

2D Example

Generating a 2D hdf5 table in python:

```
import h5py
import numpy as np

f = lambda i0, i1: -0.1 * (i0 - 300) + 1e-4*(i1 - 1e5) + 100.0
temp = np.linspace(300, 600, 4, dtype='d')
pressure = np.linspace(1e5, 5e5, 3, dtype='d')
temp2, pressure2 = np.meshgrid(temp, pressure, indexing="ij")
rho = f(temp2,pressure2)

with h5py.File("2d_example.h5", 'w') as h5File:
  h5File.create_dataset("function", data=rho)
  h5File.create_dataset("i0", data=temp)
  h5File.create_dataset("i1", data=pressure)
```

Corresponding h5dump output for the 2D example table:

```
HDF5 "2d_example.h5" {
GROUP "/" {
  DATASET "function" {
      DATATYPE  H5T_IEEE_F64LE
      DATASPACE  SIMPLE { ( 4, 3 ) / ( 4, 3 ) }
      DATA {
      (0,0): 100, 120, 140,
      (1,0): 90, 110, 130,
      (2,0): 80, 100, 120,
      (3,0): 70, 90, 110
      }
```

(continues on next page)

(continued from previous page)

```
  }
  DATASET "i0" {
      DATATYPE  H5T_IEEE_F64LE
      DATASPACE  SIMPLE { ( 4 ) / ( 4 ) }
      DATA {
      (0): 300, 400, 500, 600
      }
  }
  DATASET "i1" {
      DATATYPE  H5T_IEEE_F64LE
      DATASPACE  SIMPLE { ( 3 ) / ( 3 ) }
      DATA {
      (0): 100000, 300000, 500000
      }
  }
}
}
```

3D Example

Generating a 3D hdf5 table in python:

```
import h5py
import numpy as np
import math

def complex_radial_src(theta, z, r):
    return np.sin(theta*math.pi/180)*z + r

theta = np.linspace(0, 360, 4, dtype='d')
z = np.linspace(0, 3, 3, dtype='d')
r = np.linspace(0, 0.1, 2, dtype='d')
T, Z, R = np.meshgrid(theta, z, r, indexing="ij")
src = complex_radial_src(T,Z,R)

with h5py.File("3d_example.h5", 'w') as h5File:
  h5File.create_dataset("function", data=src)
  h5File.create_dataset("i0", data=theta)
  h5File.create_dataset("i1", data=z)
  h5File.create_dataset("i2", data=r)
```

Corresponding h5dump output for the 3D example table:

```
HDF5 "3d_example.h5" {
GROUP "/" {
  DATASET "function" {
      DATATYPE  H5T_IEEE_F64LE
      DATASPACE  SIMPLE { ( 4, 3, 2 ) / ( 4, 3, 2 ) }
      DATA {
      (0,0,0): 0, 0.1,
      (0,1,0): 0, 0.1,
      (0,2,0): 0, 0.1,
      (1,0,0): 0, 0.1,
      (1,1,0): 1.29904, 1.39904,
      (1,2,0): 2.59808, 2.69808,
      (2,0,0): 0, 0.1,
      (2,1,0): -1.29904, -1.19904,
      (2,2,0): -2.59808, -2.49808,
      (3,0,0): 0, 0.1,
      (3,1,0): -3.67394e-16, 0.1,
      (3,2,0): -7.34788e-16, 0.1
      }
  }
  DATASET "i0" {
      DATATYPE  H5T_IEEE_F64LE
      DATASPACE  SIMPLE { ( 4 ) / ( 4 ) }
      DATA {
      (0): 0, 120, 240, 360
      }
  }
  DATASET "i1" {
      DATATYPE  H5T_IEEE_F64LE
      DATASPACE  SIMPLE { ( 3 ) / ( 3 ) }
      DATA {
      (0): 0, 1.5, 3
      }
  }
  DATASET "i2" {
      DATATYPE  H5T_IEEE_F64LE
      DATASPACE  SIMPLE { ( 2 ) / ( 2 ) }
      DATA {
      (0): 0, 0.1
      }
  }
}
}
```

Example Usage

The ND_Table expression is registered as a generic expression with most properties, boundary conditions, sources, and initial conditions. The independent variables are specified in-line for each model and can correspond to time, any scalar expression (e.g. DENSITY), or any scalar component of a vector or tensor expression (e.g. GRAD_TEMPERATURE_X or COORDINATES_X). The naming of these inputs should follow the naming rules described in *General Naming Convention* when referring to properties with phases or species indexes. For example, to use the pressure in the liquid phase as an independent variable one would use `i0 = liquid_phase_pressure`.

As an example, the syntax to use such a table for density as a function of pressure and temperature is shown below.

```
density = ND_Table name = CO2_density.h5 i0 = temperature i1 = pressure
```

In the table CO2_density.h5, function must correspond to tabulated density data (2D array) while the parameters i0 and i1 specify the expression labels that the independent table variables each map to.

These tables can also be used in boundary conditions, initial conditions, sources, user expressions, and other places where generic expressions are allowed using a similar syntax:

```
BC flux for energy on surface_1 = ND_Table name = flux.h5 i0 =␣
↪coordinates_x \$
      i1 = coordinates_z i2 = time
```

```
Source for energy on all_blocks = ND_Table name = source.h5 i0 =␣
↪coordinates_z \$
      i1 = coordinates_y i2 = coordinates_x
```

Table Signature

Traceability is important and since the actual contents of the HDF5 table used for an ND_Table expression are not shown in the log file there is a summary and signature block output in the log file for each table used in the problem. This summary block shows the number of table dimensions, the size of each dataset, the min and max of each dataset, and a hash ("uid") of the full table data set. Two tables with the same data will have the same hash, regardless of their filename.

```
Read HDF5 table 'demo.h5' {
  uid: 82314a8bcdc16dec
  nDims: 2
  i0: 361 entries in range [0, 360]
  i1: 6 entries in range [-1e+06, 1e+06]
```

(continues on next page)

(continued from previous page)

```
  f: 2166 entries in range [400, 675]
}
```

User Fields

Fields in Aria are automatically created for things like solution fields and post-processed fields, but sometimes a user may need to manually create a field to *transfer* to or from. These fields can be declared in the Aria Region using the following syntax.

```
User Field REAL NODE SCALAR MyField1 on block_1
User Field REAL NODE SCALAR MyField2 on surface_1 value = 1.2
User Field REAL NODE VECTOR MyVec    on block_1   value = 1.2 1.5 0.0
```

The data type for the field can be `REAL` or `INTEGER`.

The location for the field can be `NODE`, `FACE`, or `ELEMENT`.

The rank of the field can be `SCALAR`, `VECTOR`, or `TENSOR`.

The value specification is optional, and the field will default to an initial value of 0 if omitted. For vectors or tensors, multiple values can be provided.

When creating a user field to be transferred into, one can specify a number of smoothing iterations to apply. The specified number of smoothing iterations will then be applied to the field after any time it is transferred into.

```
User Field REAL NODE SCALAR MyField1 on block_1 smooth 2 iterations
```

See the *command reference* for more information.

User Subroutines

Warning: The use of user subroutines should be considered a last resort, especially for production simulations - most custom capabilities can be implemented using GPU-compatible, well-tested capabilities like *string functions*, *user expressions*, and *multi-dimensional tables*.

While most customization can be accomplished using *String Functions*, Aria still supports the inclusion of several types of user subroutines. It is worth noting that simulations that use user subroutines will not run fully on the GPU on platforms where they are available and there may be a significant performance penalty relative to using string functions in those cases.

Aria offers support of C and Fortran user subroutines through fixed-interface signatures. Callback data query requests from the subroutines are also supported, again through fixed-interface function signatures. It is important to note that C subroutines provide some level of variable-type safety and bounds checking that is not supported with Fortran subroutines.

Usage of both C and Fortran subroutines within Aria are described in the sections which follow.

Building Usersubs

The process for building user subroutines is the same for C and Fortran subroutines. When you invoke the `sierra --make` command, the specified input file is scanned for imported shared object (`.so`) files and they are compiled from source files of the same name.

```
sierra --make aria -i input.i
```

Note: The shared object generally does not have to be re-compiled between Aria runs unless you change the contents of the C or F file, however it is a good practice to re-compile them when you change what version of Aria you are running.

Subroutines can also be compiled using the `buildsub` command, which should produce an equivalent output

```
buildsub -e aria -i input.i
```

Static Linking

Fortran subroutines also have the option of being statically linked instead using a shared object file. To use this option, include the `user subroutine file = myfile.F` line in the Aria input file (in the same scope as the `load user plugin file` command). When this option is used, a new Aria executable is created in a `UserSubsProject/bin` directory created at the path where the `sierra --make` command was invoked.

C Subroutines

A C user subroutine must be a text file of suffix .C that will be compiled to a shared object file of the same name with the `.so` extension. For example, a file named `Aucal_density.C` would be compiled to a file named `Aucal_density.so` and referenced from the Aria input file using the line command

```
load user plugin file ./Aucal_Density.so
```

Note that, since `Aucal_density.so` is a filename, its input file name is case-sensitive, and considerations must be made concerning its location path.

An example C file to provide a subroutine for density is shown below.

```
#include "Aria_Calore_User_Sub_Support.h"

int
aucal_density(
  UserQuery &                              user_query,
  ElementIds                               elemid,
  int                                      n,
  int                                      nint,
  int                                      spatial_dimension,
  CoordinateElementIntegrationPoints       coords,
  TemperatureElementIntegrationPoints      temp,
  ScalarElementIntegrationPoints           rho)
{
  //  iterate elements and integration points
  for (int j = 0; j < nint; ++j) {
    for (int i = 0; i < n; ++i) {
      rho(i,j) = 0.1;
    }
  }
  return 0;
}

RegisterFunction(ElemUserSubC, aucal_density, "bulk_density");
```

In the code above, the actual function (named `aucal_density`) is declared with the required signature for an element scalar. The function is then registered for use in Aria with the following command (which must be after the function declaration)

```
RegisterFunction(ElemUserSubC, aucal_density, "bulk_density");
```

The registration function takes three arguments:

- `ElemUserSubC` - The signature of the function being registered.
- `aucal_density` - The name of the function being registered.
- `"bulk_density"` - A name to refer to the subroutine by in the Aria input file.

Once registered, the subroutine can be referred to in Aria using the registered name and type:

```
density = calore_user_sub name = bulk_density type=element
```

Subroutine registration can also be done in a single function in the user C file, using

```
extern "C" void bulk_prop_subs()
{
  RegisterFunction(ElemUserSubC, aucal_density, "bulk_density");
}
```

When using this method, the command to load the plugin in the Aria input file must refer to this registration function (unless the registration function uses the default name of `dl_register`)

```
load user plugin file ./Aucal_Density.so using function bulk_prop_subs
```

Signatures

There are a limited number of recognized subroutine signatures supported for Aria C user functions, described in the following sections. Scaled signatures are used to combine the basic user subroutine functionality with field variable scaling. With the exception of the node signature, these signatures are used for subroutines that operate on quadrature point values. The primary differences between the different subroutine types are the signature (the type and order of arguments that the function takes) and the signature type (the first argument passed to the `RegisterFunction` call).

Node Signature

The node subroutine signature is generally used for Dirichlet boundary conditions.

```
/**
* Prototype for the nodal signature.
*
* user_query:        query function
* node_id:           array of global ids of the nodes
* num_nodes:         number of nodes in the array
* spatial_dimension: number of nodes in the array
* coordinate_array:  array of coordinates of the current nodes
* node_data_array:   array of value to be calculated
*/
int
AcalNodeUserSubC(
  UserQuery &     user_query,
  int             node_id,
  int             num_nodes,
  int             spatial_dimension,
  CoordinateNodes coordinate_array,
  DataNodes       node_data_array);
```

It would be registered in the C file using the NodeUserSubC signature argument, as shown in the following example subroutine

```
#include "Aria_Calore_User_Sub_Support.h"

int
dirichsub(
  UserQuery&       user_query,
  int              nodeid,
  int              nnode,
  int              spatial_dimension,
  CoordinateNodes  coords,
  DataNodes        temperature)
{
  for (int i = 0; i < nnode; ++i) {
    double ynode = coords(1,i);
    if ( ynode < 0.5 ) {
      temperature(i) = 1.5;
    } else {
      temperature(i) = 0.1;
    }
  }
  return 0;
}

RegisterFunction(NodeUserSubC, dirichsub, "dirich_bc");
```

A node subroutine would be invoked in the Aria input file by setting type=node

```
BC Dirichlet for Temperature on surface_4 = Calore_User_Sub name=dirich_
↪bc type=node
```

Element Signature

The element signature is used to define material properties, flux boundary conditions, and other boundary properties.

```
/**
* Prototype for the scalar element signature.
*
* user_query:          query function
* elem_ids:            pointer to the global id of the current element
* num_elements:        number of elements in the workset
```

(continues on next page)

(continued from previous page)

```
* nint:                number of gauss points
* coordinate_array:    coordinates array of the current node (3, nelem,␣
↪nint)
* temperature_array:   temperature array (nelem, nint)
* elem_data_array:     array of values to be calculated (nelem, nint)
*/
int
AcalElemUserSubC(
  UserQuery &                              user_query,
  ElementIds                               elem_ids,
  int                                      num_elements,
  int                                      nint,
  int                                      spatial_dimension,
  CoordinateElementIntegrationPoints       coordinate_array,
  TemperatureElementIntegrationPoints      temperature_array,
  ScalarElementIntegrationPoints           element_data_array)
```

It would be registered in the C file using the `ElemUserSubC` signature argument, as shown in the following example subroutine

```
#include "Aria_Calore_User_Sub_Support.h"

int
aucal_density(
  UserQuery &                              user_query,
  ElementIds                               elemid,
  int                                      nelem,
  int                                      nint,
  int                                      spatial_dimension,
  CoordinateElementIntegrationPoints       coords,
  TemperatureElementIntegrationPoints      temp,
  ScalarElementIntegrationPoints           rho)
{
  for (int j = 0; j < nint; ++j) {
    for (int i = 0; i < nelem; ++i) {
      rho(i,j) = 0.1 + temp(i,j);
    }
  }
  return 0;
}

RegisterFunction(ElemUserSubC, aucal_density, "aucal_density");
```

Scaled Element Signature

This variant of the element subroutine employs an additional field variable that is supplied through the function signature. Here the field variable is used to perform some logical decision on how the result is assigned and the name of the Field variable to be used is specified from the input file. The call signature for scaled scalar element-based user subroutines is

```
/**
* Prototype for the scaled scalar element signature.
*
* user_query:         query function
* elem_ids:           pointer to the global id of the current element
* num_elements:       number of elements in the workset
* nint:               number of gauss points
* coordinate_array:   coordinates array of the current node (3, nelem,␣
 ↪nint)
* temperature_array:  temperature array (nelem, nint)
* scaling_array:      scaling array (nelem, nint)
* elem_data_array:    array of values to be calculated (nelem, nint)
*/
int
AcalScaledElemUserSubC(
  UserQuery &                           user_query,
  ElementIds                            elem_ids,
  int                                   num_elements,
  int                                   nint,
  int                                   spatial_dimension,
  CoordinateElementIntegrationPoints    coordinate_array,
  TemperatureElementIntegrationPoints   temperature_array,
  ScalarElementIntegrationPoints        scaling_array,
  ScalarElementIntegrationPoints        element_data_array)
```

and it is registered using the `ScaledElemUserSubC` signature argument.

Scaled user subroutines are typically used in Calore block-style commands

```
begin convective flux boundary condition convect
  add surface surface_2
  reference temperature = 180.0
  scaled convective coefficient subroutine is aucal_htcoef fieldname = c0
end convective flux boundary condition convect
```

One should note that the number of arguments for the “Scaled Element Signature” and “Reference Temperature Element Signature” are identical. Attempts to use a user subroutine in the wrong context will be detected while parsing the input file.

Reference Temperature Element Signature

In some problems the surface heat transfer coefficient is determined to be a function of reference temperature rather than just the surface temperature. One example of this would be when modeling problems of natural convection where one computes a Rayleigh number based upon the difference between surface temperature and reference temperature to determine the local Nusselt number and a local heat transfer coefficient. In these cases it is sometimes more convenient to utilize an existing model for reference temperature and use those values to evaluate the heat transfer coefficient based upon additional criteria within a user subroutine. Here the interface for the Element Signature is modified to also supply interpolated values of the heat transfer coefficient at the quadrature points with the following call signature

```
/**
* Prototype for the scalar reference temperature dependent
* heat transfer coefficient signature.
*
* user_query:            query function
* elem_ids:              pointer to the global id of the current element
* num_elements:          number of elements in the workset
* nint:                  number of gauss points
* coordinate_array:      coordinates array of the current node (3, nelem,␣
 ↪nint)
* ref_temperature_array: reference temperature array (nelem, nint)
* temperature_array:     temperature array (nelem, nint)
* elem_data_array:       array of values to be calculated (nelem, nint)
*/
int
AcalElemUserSubHtC(
  UserQuery &                           user_query,
  ElementIds                            elem_ids,
  int                                   num_elements,
  int                                   nint,
  int                                   spatial_dimension,
  CoordinateElementIntegrationPoints    coordinate_array,
  ScalarElementIntegrationPoints        ref_temperature_array,
  TemperatureElementIntegrationPoints   temperature_array,
  ScalarElementIntegrationPoints        element_data_array)
```

and it is registered using the `ElemUserSubHtcC` signature argument. This would typically be used in a Calore-style convective BC

```
begin convective flux boundary condition convect
  add surface surface_2
  reference temperature = 180.0
```

(continues on next page)

(continued from previous page)

```
  ref temp convective coefficient subroutine is aucal_htcoef
end convective flux boundary condition convect
```

Scaled Reference Temperature Element Signature

The scaled version of the reference temperature element signature has an additional argument for the scale field

```
/**
* Prototype for the scaled scalar reference temperature dependent
* heat transfer coefficient signature.
*
* user_query:            query function
* elem_ids:              pointer to the global id of the current element
* num_elements:          number of elements in the workset
* nint:                  number of gauss points
* coordinate_array:      coordinates array of the current node (3, nelem,␣
 ↪nint)
* ref_temperature_array: reference temperature array (nelem, nint)
* temperature_array:     temperature array (nelem, nint)
* scale_elem_data_array: scaling value array (nelem, nint)
* elem_data_array:       array of values to be calculated (nelem, nint)
*/
int
AcalScaledElemUserSubHtcc(
  UserQuery &                           user_query,
  ElementIds                            element_ids,
  int                                   num_elements,
  int                                   num_integration_points,
  int                                   spatial_dimension,
  CoordinateElementIntegrationPoints    coordinate_array,
  ScalarElementIntegrationPoints        ref_temperature_array,
  TemperatureElementIntegrationPoints   temperature_array,
  ScalarElementIntegrationPoints        scaling_array,
  ScalarElementIntegrationPoints        flux_array)
```

and is registered using the `ScaledElemUserSubHtcC` signature type. This would typically be used in a Calore-style convective BC

```
begin convective flux boundary condition convect
  add surface surface_2
  reference temperature = 180.0
```

(continues on next page)

(continued from previous page)

```
  scaled ref_temp convective coefficient subroutine = aucal_htcoef␣
↪fieldname = c0
end convective flux boundary condition convect
```

User Data

All the subroutine signatures take a `UserQuery` object, which can be used to retrieve a variety of data from the Aria input file and simulation state. If you add a data block in the input file (at the sierra level)

```
Begin data block mydata
  real foo = (293. 433. 533. 623. 773. 1273. )
  real bar = 12.0
End
```

Then you can retrieve these values in the user subroutine using the `UserQuery` object. The data block can be "bound" to different scopes to allow access without using it's name. For example, to bind it to the region scope you would use the following command in the Aria Region.

```
Use data block mydata
```

Similarly, to bind it to an Aria material you would use that command in the Aria Material block.

A list of the common methods that can be called on a `UserQuery` object are

- `uq.currentTime()` - Return the current time
- `uq.currentTimeStepSize()` - Return the current time step size
- `uq.getUserRealMaterialData(arr, var_name)` - Retrieve a data array from a data block bound with a `Use data block` call in the Aria Material block.
- `uq.getUserRealRegionData(arr, var_name)` - Retrieve a data array from a data block bound with a `Use data block` call in the Aria Region block.
- `uq.getUserRealInstanceData(arr, var_name)` - Retrieve a data array from a data block bound with a `Use data block` call in the relevant IC/BC/Source block.
- `uq.getUserRealData(arr, var_name, data_block_name)` - Retrieve a data array from a data block not bound with a `Use data block` call, instead using the data block name directly in the function call.
- `uq.getGlobalVariable(var, name)` - Retrieve the value of a global variable by name
- `uq.getGlobalVariableValue<double>(name)` - Return a global variable value by name (specify the variable type as the template parameter)

For example, to retrieve data from a block bound to the material scope, you would use

```
RealArray1d foo(6);
RealArray1d bar(1);

user_query.getUserRealMaterialData(foo, "foo");
user_query.getUserRealMaterialData(bar, "bar");

double t = user_query.currentTime();
double p = user_query.getGlobalVariableValue("enc_pressure");
```

Fortran Subroutines

An Aria Fortran user subroutine must be a text file of suffix .F or .f, in FORTRAN77 format. The syntax for dynamically linked Fortran subroutines is the same as for C ones:

```
load user plugin file Aucal_ConvFlux.so using function conv_flux_register
```

where `conv_flux_register` is the name of the Fortran subroutine used for user subroutine registration.

For statically linked subroutines, the additional `user subroutine file = myfile.F` command should be included in the input file.

Signatures

There are only two valid Fortran subroutine signatures supported in Aria - node and element routines. A Fortran subroutine requires a registration function to be defined to register functions for use in Aria. Like the C subroutine registration, the call to `fmwkusersub` takes three arguments, the signature, the function, and a string to use to refer to it in the Aria input file.

```
c============================================================
c   Must have a registration subroutine
c
      subroutine material_register

      implicit none
      external aucal_fort_elem_sub_f
      external my_rho

      call fmwkusersub(aucal_fort_elem_sub_f, my_rho,
     &                 'my_rho')

      end
```

(continues on next page)

(continued from previous page)

```
c=============================================================
c    Actual density calculation
c
      subroutine my_rho(objid,nelem,nint,coords,temperature,
     &                  rho,time,rdat,idat,ierror)

      implicit none
      integer nelem,nint
      integer objid(nelem)
      double precision coords(3,nelem,nint)
      double precision temperature(nelem,nint)
      double precision rho(nelem,nint)
      double precision time
      double precision rdat
      double precision idat
      integer ierror

      integer i, j

      ierror = 0

      do i=1,nelem
        do j=1,nint
          rho(i,j) = 0.1 + 0.001*temperature(i,j)
        end do
      end do

      return
      end
```

This registration function is then referred to in the Aria input file using

```
load user plugin file mysub.so using function material_register
```

and the subroutine can be referenced in a material property as

```
density = Fortran sub_name = my_rho
```

Node Signature

An example nodal Fortran routine is shown below - for applying a Dirichlet BC.

```
c
c     Subroutine to handle coordinate and time-dependent
c       Temperature Dirichlet BC
c
      subroutine dirich_test(objid,nnodes,coords,temperature,
     &                       time,rdat,idat,ierror)
      implicit none
      integer nnodes
      integer objid
      double precision coords(2,nnodes)
      double precision temperature(nnodes)
      double precision time
      integer ierror
      double precision rdat(1)
      integer idat(1)
c
c    INPUT ARGUMENTS:
c        objid(nelem): array of mesh object IDs.
c        nelem: number of elements in the workset
c        nint:  Number of integration points per element
c        coords(2,n,nint): array of the coordinates (x,y,z)
c                          of the Gauss points
c        rdat() real parameter data
c        idat() integer data
c
c    OUTPUT ARGUMENTS:
c        temperature: temperature at the node
c        ierror: user defined error code,
c                zero means success.
c
      integer i, j

      ierror = 0

      do i=1,nnodes

        if( coords(2,i) .lt. 0.5 ) then
          temperature(i) = 1.5
        else
          temperature(i) = 0.1
```

(continues on next page)

(continued from previous page)

```
        end if

      end do

      return
      end
```

The command to use this in the Aria input file would be

```
Begin Temperature Boundary Condition bcblock1
  Temperature fortran subroutine = dirich_test
  Add surface surface_2
End Temperature Boundary Condition bcblock1
```

Element Signature

A Fortran element subroutine can be used to apply a flux, define a source, or define material properties. The example below shows how to use it to define a heat flux boundary condition

```
c
c     Subroutine to handle coordinate, time and temperature
c        dependent flux
c
      subroutine flux_test(objid,nelem,nint,coords,temperature,
     &                     flux,time,rdat,idat,ierror)
      implicit none
      integer nelem, nint
      integer objid(nelem)
      double precision coords(3,nelem,nint)
      double precision temperature(nelem,nint)
      double precision flux(nelem,nint)
      double precision time
      integer ierror
      double precision rdat(2)
      integer idat(3)
c
c    INPUT ARGUMENTS:
c        objid(nelem): array of mesh object ID's.
c        nelem: number of elements in the workset
c        nint:  Number of integration points per element in the workset
c        coords(3,n,nint): array of the coordinates (x,y,z) of the Gauss␣
 ↪points
```

(continues on next page)

(continued from previous page)

```
c        temperature(n,nint): temperature at the Gauss points
c        rdat() real parameter data
c        idat() integer data
c
c    OUTPUT ARGUMENTS:
c        flux:  array of length ( n, nint)  containing
c                the heat flux coefficient
c                at integration points
c        ierror: user defined error code for calore to test.
c                zero means success.
c
      integer i, j

      ierror = 0

      do i=1,nelem

        do j=1,nint
            if( coords(3,i,j) .gt. 4.0 ) then
              flux(i,j) = 4000.0
            else
              flux(i,j) = 0.0
            end if

        end do

      end do

      return
      end
```

This is referenced in the Aria input file using the following commands

```
Begin Heat Flux Boundary Condition C1
  Add Surface surface_1
  Flux Fortran Subroutine = ht_flux_test
  integer data 8, 12, 14
  real data 1.0 2.0
end
```

User Plugins

Warning: The use of user plugins should be considered a last resort, especially for production simulations - most custom capabilities can be implemented using GPU-compatible, well-tested capabilities like *string functions*, *user expressions*, and *multi-dimensional tables*.

User plugins are another way that users can extend Aria's capabilities using custom compiled code. Unlike user subroutines, which are functions with a fixed signature called to evaluate certain properties or terms, plugins are C++ classes that extend the `Expression` class and can be used interchangeably with native Aria expressions. Before making a plugin, it is recommended that a user reviews the *Expression Overview*.

Since Aria uses a Newton formulation, sensitivities are required from each expression. There are three ways of providing sensitivities in an expression:

1. Manually code them
2. Use numerical finite-differences
3. Use Forward Automatic Differentiation (FAD)

The FAD method is ideal for plugins because the sensitivities are performed analytically and are exact, though there may be a small performance degradation relative to manually coded sensitivities. For that reason, we recommend using the FAD approach when writing plugins.

Writing a Plugin

In this section we'll write an example plugin to calculate density as $\rho = a + bT + cT^2 + dT^3$. The first part of our plugin C file will define the plugin class.

```
#include <Aria_Plugin_Expression.h>
using namespace sierra::Aria;

class My_Density : public FAD_Expression
{
public:
  My_Density(Expression_Manager * const manager,
             const sierra::Identifier & expr_model_name,
             const Subdomain_Tag & subdomain_tag,
             const int & subindex,
             const Phase_Label & phase_label,
             const PrsrParamMap& params);
  virtual ~My_Density() {}
```

(continues on next page)

(continued from previous page)

```
  virtual void compute_FAD_values(const ElementRange & elem_range);
private:
  double a = 0.0;
  double b = 0.0;
  double c = 0.0;
  double d = 0.0;

  Scalar_FAD_Expression_Handle values;
  Scalar_FAD_Expression_Handle temperature;
};
```

There are a few things to note in this definition

- The class inherits from `FAD_Expression` so it does not require a `compute_sensitivities` method.
- The arguments to the constructor must be as-shown here.
- You must define expression handles for the value being computed and any prerequisites.

The next part of the plugin C file is the constructor definition. This is where you read in any user-supplied parameters needed for your expression, define what expression your model provides (`EXPR::DENSITY` here), define the rank (tensor order) of the provided value, and define any prerequisites your expression needs (`EXPR::TEMPERATURE` here).

```
My_Density::My_Density(Expression_Manager * const manager,
                       const sierra::Identifier & expr_model_name,
                       const Subdomain_Tag & subdomain_tag,
                       const Int & subindex,
                       const Phase_Label & phase_label,
                       const PrsrParamMap& params)
: FAD_Expression(manager,subdomain_tag,subindex,phase_label)
{
  my_tensor_order          = 0;
  my_expression_tag        = Expression_Tag(EXPR::DENSITY,NO_OP,subindex,
↪phase_label);
  my_expression_model_name = expr_model_name;

  // List the expressions that are required for this model.
  const Expression_Tag temperature_tag(EXPR::TEMPERATURE,NO_OP,subindex,
↪phase_label);

  add_prereq(temperature_tag,temperature);

  // Get my model parameters.
```

(continues on next page)

(continued from previous page)

```
  get_optional_parameter(params,"A",a);
  get_optional_parameter(params,"B",b);
  get_optional_parameter(params,"C",c);
  get_optional_parameter(params,"D",d);

  if(a == 0.0 && b == 0.0 && c == 0.0 && d == 0.0)
  {
    throw sierra::RuntimeUserError() << "ERROR: All MY_MODEL_DENSITY␣
↪parameters are zero.";
  }

  // Make myself known to the manager.
  register_myself(values);
}
```

The next component of the plugin C file is the `compute_FAD_values` method, where the actual density value is computed

```
void My_Density::compute_FAD_values(const ElementRange & elem_range)
{
  for (auto elem : elem_range)
  {
    for(auto point : my_sizes.get_points())
    {
      const auto& T = temperature(elem,point);
      values(elem,point)  = a + T*(b + T*(c + T*(d)));
    }
  }
}
```

Note that since our Expression is a scalar in this example, the signature of `values` is `values(elem,point)`. If our result was a vector or tensor, the signature would be `values(elem,point,r)` and `values(elem,point,r,s)`, respectively.

The last thing we need to do is register the plugin

```
ExprPluginFactory<My_Density> my_density_creator("MY_MODEL_DENSITY");
```

The name provided here will define how we refer to the plugin in the input file (`MY_MODEL`) and what property it is valid for (`DENSITY`).

Note: The naming of the plugin here must follow the pattern of `SOMENAME_PROPERTYNAME` where `PROPERTYNAME` defines what it can be used for in the input file. For example `BEST_MODEL_DENSITY` or `STUPENDOUS_ENERGY_FLUX` could be referred to in the input file using

```
density = user_plugin name=best_model
BC flux for energy on surface_1 = user_plugin name=stupendous
```

Compiling a Plugin

User plugins are compiled to shared object (`.so`) files using the same procedure as *User Subroutines*. The following command will scan the input file for import `.so` files and compile them from source files of the same name.

```
sierra --make aria -i input.i
```

Using a Plugin

In the Aria input file, the plugin is loaded using the command

```
Begin Sierra My_Sierra_Job
   Load User Plugin file ./My_Model_Plugins.so
```

To use a plugin for a material property, source, or flux you use the following syntax, where the name of the plugin (e.g. `My_Model`, `My_Source`, or `My_Flux`) was defined in your C file. Any parameters needed for your plugin are supplied in-line (e.g. `a=1 b=-0.01`).

```
Begin Aria Material Kryptonite
  Density = User_Plugin name=My_Model a=1.0 b=-0.01
  ...
End

...

Begin Aria Region Region
  ...
  Source for Energy on block_1 = User_Plugin name=My_Source

  BC flux for Energy on surface_1 = User_Plugin name=My_Flux d=5.
  ...
End
```

Debugging a Plugin

Many times, you can check that your plugin is functioning properly by *Postprocessing* the value your plugin is computing and verifying that the values are as expected. However, if this is not sufficient there are a few other techniques you can use.

If you have coded sensitivities manually, you can use the `--tftklog sens_check` command line option to verify that your analytical sensitivities match numerical finite difference sensitivities.

You can also add logging statements in the plugin to check values directly, using the `arialog.m(LOG_PLUGIN)` command.

```
void My_Density::compute_FAD_values(const ElementRange & elem_range)
{
  for (auto elem : elem_range)
  {
    for(auto point : my_sizes.get_points())
    {
      const auto& T = temperature(elem,point);
      values(elem,point)  = a + T*(b + T*(c + T*(d)));
      arialog.m(LOG_PLUGIN) << "value(" << elem << "," << point << ") = "
↪<< value(elem,point)
                            << " when T = " << T << Diag::endl;
    }
  }
}
```

The `LOG_PLUGIN` mask means this log stream is only activated when you run the code with `--arialog plugin`. There's a performance penalty for having this logging code option present (even if you don't turn the logging on) so you probably want to remove it once you're done debugging.

User Utilities

Warning: The use of user utilities should be considered a last resort, especially for production simulations - most custom capabilities can be implemented using GPU-compatible, well-tested capabilities like *string functions*, *user expressions*, and *multi-dimensional tables*.

User utilities are another way that users can extend Aria's capabilities using custom compiled code. Unlike a *plugin expression*, which is evaluated locally at different nodes or integration points in an expression graph controlled by Aria, a user utility is a standalone class with an `execute` function that can be executed at various points during problem execution. A plugin

expression cannot generally access field data or perform MPI calls, but a user utility can do both of those things.

The tradeoff is that the API for a user utility is much more general and open, and as such cannot be completely documented here. This section will provide some guidelines on how to define them, and some example utilities. For more advanced capabilities, you will likely need to involve an Aria developer.

Writing a Utility

In this section we'll write an example utility to calculate a custom field we will use for a boundary flux. The first part of our plugin C file will define the utility class. The API requirements for this class are:

1. It must inherit from `User_Equation_System_Utility`
2. Its constructor must use the following signature - `MyClass(Equation_System & equation_system, const PrsrParamMap & params)`
3. It must define an execute function as `void execute() override`
4. It may define an initialization function as `void initialize() override`, but this is not required

Our class definition looks like:

```
#include <stk_mesh/base/Selector.hpp>
#include <eq_utilities/Aria_EquationSystemUtilityFactory.h>
#include <Aria_Equation_System.h>
#include <Aria_Plugin_Expression.h>
#include <Aria_DiagWriter.h>
#include <Aria_Mesh.h>
#include <tftk_mesh/tftk_FieldFunctions.h>

using namespace sierra::Aria;

class MyCustomFluxComputer : public User_Equation_System_Utility
{
public:
  MyCustomFluxComputer(
    Equation_System & equation_system,
    const PrsrParamMap & params
  );

  ~MyCustomFluxComputer() override = default;
```

(continues on next page)

(continued from previous page)

```
  void initialize() override;
  void execute() override;

private:
  double compute_flux(double tN, double tNP1, double x, double y, double␣
 ↪z);

  Region & my_region;
  String my_flux_field_name;
  double my_update_interval;
  tftk::mesh::FieldRef my_coordinates;
  tftk::mesh::FieldRef my_flux;
};
```

The next part of the utility C file is the constructor definition. This is where you read in any user-supplied parameters needed for your utility and define where in the execution sequence it should be run. Common choices for where the utility should be run are:

- `RUN_INITIALLY` - Run only once during initialization (before the first time step)
- `RUN_ONCE` - Run only once at the first time step
- `BEGIN_TIMESTEP` - Run at the start of each time step before any equation systems begin
- `PRE_NONLINEAR_SOLVE` - Run before beginning nonlinear iteration at each time step
- `POST_NONLINEAR_SOLVE` - Run after completing nonlinear iteration at each time step
- `PRE_ITERATE` - Run before each nonlinear Newton solve iteration (before matrix assembly)
- `POST_ITERATE` - Run at the end of each nonlinear Newton solve iteration

Additionally, when you define the utility in your aria input file you can pass any number of key-value pairs to it, which will be available in the `PrsrParamMap` object passed in here. For example, the case below is called in the Aria input file with

```
Add Utility MyFluxUtil dt=1 flux_field=my_flux
```

In the code, we can then get the values passed in for "dt" and "flux_field" and save them in our utility for later use. Additional methods are available for handling optional parameters as well.

```
MyCustomFluxComputer::MyCustomFluxComputer(
  Equation_System & equation_system,
  const PrsrParamMap & params
)
: User_Equation_System_Utility(
    equation_system,
```

(continues on next page)

(continued from previous page)

```
    PRE_NONLINEAR_SOLVE,
    "MyCustomFluxComputer"
  )
, my_region(*(equation_system.get_region()))
{
  my_flux_field_name = get_required_parameter<String>(params, "flux_field
↪", ParamCtx{"MyCustomFluxComputer"});
  my_update_interval = get_required_parameter<double>(params, "dt",␣
↪ParamCtx{"MyCustomFluxComputer"});
}
```

The next components of the plugin C file are the definitions for the other functions in the utility

```
void MyCustomFluxComputer::initialize()
{
  arialog << "Initializing MyCustomFluxComputer..." << Diag::dendl;
  auto& mesh = my_region.mesh();
  my_coordinates = mesh.get_node_field("model_coordinates");
  my_flux = mesh.get_node_field(my_flux_field_name);
}

void MyCustomFluxComputer::execute()
{
  arialog << "MyCustomFluxComputer setting flux field" << Diag::dendl;

  const double tN = my_region.time(STATE_N);
  const double tNP1 = my_region.time(STATE_NP1);

  arialog << "  TN = " << tN << ", TNP1 = " << tNP1 << Diag::dendl;

  auto& mesh = my_region.mesh();
  auto selector = mesh.active_not_aura_selector() &␣
↪stk::mesh::selectField(my_flux);

  using F = tftk::mesh::EntityValues<double>;
  using ConstF = tftk::mesh::EntityValues<const double>;

  tftk::mesh::for_each_entity_run(
    "MyFluxCalculator",
    mesh,
    selector,
    my_flux,
    {my_coordinates},
```

(continues on next page)

(continued from previous page)

```
    KOKKOS_LAMBDA(const F& flux, const ConstF& xyz) {
      flux() = -compute_flux(tN, tNP1, xyz(0_comp), xyz(1_comp), xyz(2_
↪comp));
    },
    tftk::HostSpace()
  );
}

double MyCustomFluxComputer::compute_flux(double tN, double tNP1, double␣
↪x, double y, double z) {
  return tNP1;
}
```

Finally, we need to register the utility. This is done with the following command:

```
EquationSystemUtilityFactory<MyCustomFluxComputer> init_flux_computer(
↪"MyFluxUtil");
```

The name used here ("MyFluxUtil") is the name you will use in the Aria input file to activate this utility. The name of the variable here (`init_flux_computer`) does not matter.

Compiling a Utility

User utilities are compiled to shared object (`.so`) files using the same procedure as *User Subroutines* and *User Plugins*. The following command will scan the input file for import `.so` files and compile them from source files of the same name.

```
sierra --make aria -i input.i
```

Using a Utility

In the Aria input file, the plugin is loaded using the command (assuming the plugin is defined in `My_Utility_Plugin.C`)

```
Begin Sierra My_Sierra_Job
   Load User Plugin file ./My_Utility_Plugin.so
```

Inside the Aria region, we activate it with the `Add Utility` command. The example below shows how we can define a nodal field, activate our utility to populate the field, and use the field in a flux boundary condition.

```
Begin Aria Region Region
  ...
  User field real node scalar my_flux on surface_1 value = 0
  Add Utility MyFluxUtil dt=1 flux_field=my_flux
  BC Flux for Energy on surface_1 = user_field name = my_flux
  ...
End
```

Debugging a Utility

Because of the broad, nearly unlimited access a utility provides to Aria internal code mechanics it can be difficult to write and debug a utility without access to and understanding of the broader Aria codebase, or at least other STK-based codes. If you encounter difficulties getting a utility working, we recommend contacting sierra-help@sandia.gov for assistance. Adding logging to your utility with the `arialog` command is recommended as a way of checking that it is functioning as expected as well.

4.6 Postprocessing

Table of Contents

- *Overview*
- *Fluxes*
  - *Integrated Flux*
  - *Average Flux*
  - *Flux Fields*
- *Standard Reductions (integral, average, etc)*
  - *Operator Definitions*
- *Aliased Reduction Operations*
- *Field Values*
- *Data Probes*
  - *Points outside the domain*
- *Mesh Fields*
- *Global Values*

- *Post-Processing Specific Models*
- *Universal Expressions*
- *Postprocessing Existing Results*
- *Advanced Details*
  - *Selecting Post-Processing Domains*
  - *Block Style Commands*
  - *Self Referencing Field Postprocessing*
  - *Binary State Fields*
  - *Nodal Averaging Method*
- *Legacy Postprocessor Conversion*

Overview

Aria supports a number of different post-processors for doing in-situ post-processing of custom and pre-defined quantities. The native Aria post-processors are built on the same expression system that the rest of Aria uses and are compatible with GPU execution and meshes over the 32-bit limit. An *expression* in Aria is defined either as a material property, like `DENSITY` here:

```
DENSITY = [MODEL_NAME]
```

or as a degree of freedom (DOF) solved for by an equation, like `TEMPERATURE` here:

```
EQ ENERGY FOR TEMPERATURE ON BLOCK_1 USING DIFF
```

See *Expression Overview* for more information on *expressions*.

The Aria post-processors let you calculate exact fluxes for equations, or perform common reductions and other operations on these expressions. You can also use the expressions as inputs to user-defined string functions to produce custom output. Refer to *String Functions* for more details about constructing string functions.

The general syntax for Aria post-processors is

```
POSTPROCESS [OP] OF [TARGET] [NAME] ON [LOCATION] AS [OUTPUT]
```

The following sections show the specific arguments available to use and how to apply this command to a variety of common post-processing operations. For a comprehensive listing of all possible commands, the user is directed to *Postprocessing*.

Note: In problems contained element death, post-processed reductions represent the values just before element death.

Note: Some post-processing commands are shown here on multiple lines using the \$ line continuation character. This is optional - they can be on one line if you prefer.

Fluxes

Integrated Flux

Flux post-processors can be constructed in two ways. The most accurate method is to use the residual of a given equation on the boundary nodes excluding the boundary condition contributions. As long as the equation is solved to a sufficiently small linear residual, this method will return the exact flux applied by the discretized governing equations.

Note: Since the residual flux approach works by skipping boundary condition contributions it will not work correctly on sidesets that do not have a boundary condition applied (e.g. internal sidesets). Additionally, in cases with shell surfaces that have different flux boundary conditions on each side of the shell, the integrated flux will compute the total flux on the shell surface, not the individual contributions. In both of these cases, an expression flux postprocessor should be used.

The syntax to use the residual approach requires specifying an equation name (EQ), sideset name (SURF), and output variable name (OUTPUT):

```
POSTPROCESS INTEGRATED_FLUX OF EQUATION [EQ] ON [SURF] AS [OUTPUT]
```

For example, to post-process the heat flux on surface_1 one could use:

```
POSTPROCESS INTEGRATED_FLUX OF EQUATION ENERGY ON SURFACE_1 AS HEAT_FLUX
```

The result of this operation will be a global variable named HEAT_FLUX, which follows the Aria sign convention where positive flux is out of the domain. Since the flux (W/m^2 in SI-MKS) is integrated over the specified surface, the result will have units of W if the problem is in SI-MKS units.

In addition to post-processing a flux using the nodal residual of an equation, one can also post-process a flux of a given vector expression (EXPR).

```
POSTPROCESS INTEGRATED_FLUX OF EXPRESSION [EXPR] ON [SURF] AS [OUTPUT]
```

For example, to use the HEAT_CONDUCTION expression (a flux vector defined as $-k\nabla T$) to post-process heat flux one could use:

```
POSTPROCESS INTEGRATED_FLUX OF EXPRESSION HEAT_CONDUCTION ON SURFACE_1 AS␣
↪HEAT_FLUX
```

Note: Calculating integrated flux using an expression vector like HEAT_CONDUCTION will include discretization errors not present in the residual approach and may be significantly less accurate. The residual approach should be preferred where possible. See *Expression Flux Limitations* for more discussion on this point.

A user-defined vector function can also be provided for the flux integration if desired, for example:

```
POSTPROCESS INTEGRATED_FLUX OF VECTOR_STRING_FUNCTION    \$
  f_x = "-heat_conduction_x + density*velocity_x"        \$
  f_y = "-heat_conduction_y + density*velocity_y"        \$
  f_z = "-heat_conduction_z + density*velocity_z"        \$
  ON SURFACE_1 AS HEAT_MASS_FLUX
```

Warning: Expression flux postprocessors use the outward facing normal of the domain on which they are run. For this reason, care must be taken when running an expression flux postprocessor on an internal sideset. In these cases, the reduction will contain contributions from both sides of the shared interface, such that terms from the +normal and -normal sides will (partially) cancel. It is recommended to always specify a *touching block* when postprocessing on internal sidesets to clearly define the normal vector e.g.

```
POSTPROCESS INTEGRATED_FLUX OF EXPRESSION HEAT_CONDUCTION ON \$
  INTERNAL_SURFACE TOUCHING BLOCK_1 AS HEAT_FLUX_B1
```

If the average across the internal sideset is desired, a `global_function` can be used (noting the minus sign on one term) e.g.

```
POSTPROCESS INTEGRATED_FLUX OF EXPRESSION HEAT_CONDUCTION ON \$
  INTERNAL_SURFACE TOUCHING BLOCK_1 AS HEAT_FLUX_B1
POSTPROCESS INTEGRATED_FLUX OF EXPRESSION HEAT_CONDUCTION ON \$
  INTERNAL_SURFACE TOUCHING BLOCK_2 AS HEAT_FLUX_B2
POSTPROCESS GLOBAL_FUNCTION "(HEAT_FLUX_B1 - HEAT_FLUX_B2)/2"␣
↪AS HEAT_FLUX_B12
```

Average Flux

Similar to the integrated flux command, one can also post-process an average flux using either an equation residual or a vector expression (or string function):

```
POSTPROCESS AVERAGED_FLUX OF EQUATION [EQ] ON [SURF] AS [OUTPUT]
POSTPROCESS AVERAGED_FLUX OF EXPRESSION [EXPR] ON [SURF] AS [OUTPUT]
```

The averaged flux is defined as the integrated flux over the specified surface divided by the area of the specified surface. The resulting global variable will have units of W/m^2 for a problem in SI-MKS units and the same caveats about accuracy of the expression approach apply.

The `AVERAGED_FLUX` reduction also allows an optional weighting factor for doing a weighted average (only when operating on an expression or string function, not using the residual-based flux on an equation). For example:

```
POSTPROCESS AVERAGED_FLUX OF EXPRESSION HEAT_CONDUCTION \$
   ON SURFACE_1 WITH WEIGHTING "DENSITY" AS WFLUX
```

Flux Fields

In addition to the integral and average flux reductions described in the prior sections you can also output the residual flux as either a scalar or vector field.

```
POSTPROCESS FLUX_SCALAR_FIELD OF EQUATION [EQ] ON [SURF] AS [OUTPUT]
POSTPROCESS FLUX_VECTOR_FIELD OF EQUATION [EQ] ON [SURF] AS [OUTPUT]
```

Both of these commands will create a nodal field on the specified sideset that can be included in your Exodus output file (see *Output Reference*).

The scalar field option uses the Aria sign convention where flux out of the domain is positive. The vector field option is simply the scalar field flux value multiplied by the nodal outward facing normal vector.

Standard Reductions (integral, average, etc)

Aria has several reduction types available that can take a spatially varying quantity and output a global variable:

```
POSTPROCESS [REDUCTION_TYPE] OF EXPRESSION [EXPR] ON [PART] AS [OUTPUT]
POSTPROCESS [REDUCTION_TYPE] OF FUNCTION "[FUNCTION]" ON [PART] AS␣
 ↪[OUTPUT]
```

Most of these reductions can be calculated either on volumes (blocks) or surfaces (sidesets).

Reduction types include `INTEGRAL`, `AVERAGE`, `MIN`, `MAX`, and several others. The complete list of options and details of these types are defined in *Operator Definitions*.

The spatially varying input for the reduction post-processors can be either an expression (e.g. `DENSITY` or `TEMPERATURE`) or a string function of several expressions. For example:

```
POSTPROCESS INTEGRAL OF FUNCTION "TEMPERATURE*SPECIFIC_HEAT*DENSITY" \$
     ON BLOCK_1 AS INTERNAL_ENERGY
```

For the averaging reduction types (`AVERAGE`, `AVERAGED_FLUX`, and `RMS`) one can also specify an optional weighting factor as a string function. For example, to calculate a density weighted average temperature one would use:

```
POSTPROCESS AVERAGE OF EXPRESSION TEMPERATURE ON BLOCK_1 \$
    WITH WEIGHTING "DENSITY" as Tavg
```

Operator Definitions

The available reduction operations and equations for each reduction type are defined in the following section.

Integral

The integral postprocessor reduction over a volume (block) is defined as:

$$\int_{\mathrm{V}} f(\boldsymbol{x})\,\mathrm{dV}$$

When evaluated on a surface, it is defined as:

$$\int_{\mathrm{S}} f(\boldsymbol{x})\,\mathrm{dS}$$

Average

Average (with default weight $w_e = 1$ unless specified) evaluated over a volume (blocks) is defined as:

$$\frac{\int\limits_{\mathrm{V}} f(\boldsymbol{x}) w_e(\boldsymbol{x}) \, \mathrm{dV}}{\int\limits_{\mathrm{V}} w_e(\boldsymbol{x}) \, \mathrm{dV}}$$

When evaluated on a sideset, the average is defined over the surface area as:

$$\frac{\int\limits_{\mathrm{S}} f(\boldsymbol{x}) w_e(\boldsymbol{x}) \, \mathrm{dS}}{\int\limits_{\mathrm{S}} w_e(\boldsymbol{x}) \, \mathrm{dS}}$$

RMS

RMS (with default weight $w_e = 1$ unless specified):

$$\left(\frac{\int\limits_{\mathrm{V}} |f(\boldsymbol{x})|^2 w_e(\boldsymbol{x}) \, \mathrm{dV}}{\int\limits_{\mathrm{V}} w_e(\boldsymbol{x}) \, \mathrm{dV}} \right)^{\frac{1}{2}}$$

L-1 Norm

The L-1 Norm (command `L_1_NORM`) is evaluated with:

$$\int\limits_{\mathrm{V}} |f(\boldsymbol{x})| \, \mathrm{dV}$$

L-2 Norm

The L-2 Norm (command `L_2_NORM`) is evaluated with:

$$\left(\int\limits_{\mathrm{V}} |f(\boldsymbol{x})|^2 \, \mathrm{dV} \right)^{\frac{1}{2}}$$

L-Infinity Norm

The L-∞ Norm (command `L_INF_NORM`) is evaluated with:

$$\max_{node} |f(\boldsymbol{x}_{node})|$$

H-1 Semi-Norm

The H-1 Semi-Norm (command `H_1_NORM`) is evaluated with:

$$\left(\int_{\mathrm{V}} \boldsymbol{\nabla} f(\boldsymbol{x}) \cdot \boldsymbol{\nabla} f(\boldsymbol{x}) \, \mathrm{dV} \right)^{\frac{1}{2}}$$

Max

The maximum (command `MAX`) evaluates the nodal maximum:

$$\max_{node} f(\boldsymbol{x}_{node})$$

Min

The minimum (command `MIN`) evaluates the nodal minimum:

$$\min_{node} f(\boldsymbol{x}_{node})$$

Aliased Reduction Operations

You can post-process the mass or volume of a given block using the previously defined post-processors by postprocessing an integral of density for mass or of a function of `"1"` for volume, but for convenience there are also shortcuts for post-processing mass and volume.

```
POSTPROCESS MASS ON BLOCK_1 AS M
POSTPROCESS VOLUME ON BLOCK_2 AS V
```

You can also use the `EACH_BLOCK` shortcut to automatically output these quantities per block. See *Selecting Post-Processing Domains* for more details.

Field Values

In addition to fluxes and reduction operations, you can post-process spatially varying quantities and output them as nodal fields. The syntax for this is similar to the reduction types, the primary difference is that the output is a nodal field instead of a global variable. The nodal field created by this command can then be included in your Exodus output file (see *Output Reference*).

```
POSTPROCESS VALUE OF EXPRESSION [EXPR] ON [PART] AS [OUTPUT]
POSTPROCESS VALUE OF FUNCTION "[FUNCTION]" ON [PART] AS [OUTPUT]

POSTPROCESS VALUE OF VECTOR_STRING_FUNCTION                        \$
  f_x = "[FX]" f_y = "[FY]" f_z = "[FZ]" ON [PART] AS [OUTPUT]

POSTPROCESS VALUE OF TENSOR_STRING_FUNCTION                         \$
  f_xx = "[FXX]" f_xy = "[FXY]" f_xz = "[FXZ]"                      \$
  f_yx = "[FYX]" f_yy = "[FYY]" f_yz = "[FYZ]"                      \$
  f_zx = "[FZX]" f_zy = "[FZY]" f_zz = "[FZZ]"                      \$
  ON [PART] AS [OUTPUT]
```

The OUTPUT expression is then also available for use in subsequent postprocessors. If PART is a block(s), the expression is also provided on all surfaces touching the block. For example:

```
POSTPROCESS VALUE OF FUNCTION "X*Y*Z" ON ALL_BLOCKS AS MY_EXPR
POSTPROCESS MAX OF EXPRESSION MY_EXPR ON BLOCK_1 AS MAX_EXPR_B1
POSTPROCESS MAX OF EXPRESSION MY_EXPR ON SURFACE_1 AS MAX_EXPR_S1
```

There is also a shorter syntax available for post-processing expressions:

```
POSTPROCESS [EXPR] ON [PART]
```

which is equivalent to the POSTPROCESS VALUE OF EXPRESSION command.

Data Probes

To evaluate the value of a quantity at a specified point, you can use the data probe processing command:

```
POSTPROCESS VALUE of [SRC] at [LOC] as [OUTPUT]
```

The evaluation location LOC can be specified either as - NODE <nodeID> – ID of a specific node in the mesh - POINT <x_coord> <y_coord> <z_coord> – specific location in the mesh

The supported SRC types are: - FIELD [NAME] – evaluate probe on existing field NAME - EXPRESSION [NAME] – evaluate probe on expression NAME - FUNCTION, VECTOR_STRING_FUNCTION, TENSOR_STRING_FUNCTION – evaluate probe on string function

Along with existing fields (e.g. `solution->` fields), postprocessed fields are also available for data probes. For example:

```
POSTPROCESS VALUE of EXPRESSION TEMPERATURE at POINT 0. 0. 0. as Torigin
```

is exactly equivalent to:

```
POSTPROCESS VALUE of EXPRESSION TEMPERATURE on ALL_BLOCKS as TEMP_FIELD
POSTPROCESS VALUE of FIELD TEMP_FIELD at POINT 0. 0. 0. as Torigin
```

Similarly:

```
POSTPROCESS VALUE of FUNCTION "2*TEMPERATURE" at NODE 123 as T2origin
```

is exactly equivalent to:

```
POSTPROCESS VALUE of FUNCTION "2*TEMPERATURE" on ALL_BLOCKS as DBL_TEMP_
↪FIELD
POSTPROCESS VALUE of FIELD DBL_TEMP_FIELD at NODE 123 as T2origin
```

For the `POINT` location type, the parametric tolerance and part restrictions of the search can be optionally specified using

```
POSTPROCESS VALUE of EXPRESSION TEMPERATURE at POINT 0. 0. 0. as T2origin␣
↪\$
  with TOLERANCE 1e-3 RESTRICTED to BLOCK_1 BLOCK_2
```

Warning: The `POINT` location is Eulerian (remains at a fixed point throughout the simulation), whereas the `NODE` specification is Lagrangian (sticks to that node and thus follows any deformation of the mesh at that node). For a Lagrangian specification of a specific point, see the *nearest* functionality of a heartbeat block.

Points outside the domain

By default, the `POINT` probes will pick a tolerance based on the mesh sizes and search for a matching element not farther than that tolerance. A user can also specify a tolerance with the `with tolerance X` argument.

If the probe does not find a matching element within its tolerance, the default behavior is to error out. When the point is outside any elements, but less than the tolerance away from one, it will snap to the nearest element and warn.

To change the behavior of the data probe when it is more than the tolerance outside of an element, add the `points_outside_tol` argument. For example:

```
POSTPROCESS VALUE of EXPRESSION TEMPERATURE at POINT 0. 0. 0. as T2origin␣
↪\$
  points_outside_tol = nan
```

When the point is outside the specified tolerance, its value here will be set to NaN. Other options include `zero`, `bignum` (1e100), or `abort` (the default). Using NaN should be preferred since most plotting tools will handle automatically omitting the values and it is the least likely to be mistaken for a real value.

If a data probe leaves the mesh and switches to a specified value mode it will continue to set that value every step until it comes back into the mesh, at which time it will return to normal operation.

Mesh Fields

A limited set of reduction post processors are available for post processing mesh fields directly. These are: min, max, average, L1 Norm, L2 Norm, L ∞ Norm, and RMS. Unlike expression post processors, mesh field post processors operate directly on the values stored in STK mesh fields as opposed to calculated by the expression system. The advantage of this is that it allows post processing operations that would be impossible otherwise (eg. post processing on node sets). The downside is a potential loss of accuracy as field post processors are unaware of the underlying finite element discretization and expression relationships. The most obvious example of this is that calculating the L2 norm of a solution field will give a different result than if the L2 norm of the same solution DOF were post processed using an expression based post processor. This is because the expression post processor will perform an integral using the finite element basis, whereas the field post processor will compute a simple nodal sum of squares. It is recommended that expression post processors be preferred to field post processors whenever possible.

The field post processor line command is:

```
POSTPROCESS [REDUCTION] of field [FIELD NAME] at [LOC] as [OUTPUT]
```

Field post processors can be calculated with block style syntax as well with this source line:

```
SOURCE TYPE = FIELD NAME = [FIELD NAME]
```

The reduction types that can be used with field post processors are:

- `MIN` – $\min_e f(e)$
- `MAX` – $\max_e f(e)$
- `AVERAGE` – $\frac{\sum_e f(e)}{N_e}$
- `L_1_NORM` – $\sum_e |f(e)|$
- `L_2_NORM` – $\sqrt{\sum_e f^2(e)}$

- L_INF_NORM – $\max_e |f(e)|$
- RMS – $\sqrt{\frac{\sum_e f^2(e)}{N_e}}$

e denotes the entity type over which the field is defined (element or node), N_e is the number of entities in the part the post processor is calculated over, and *f(e)* is the value of the field at a given entity.

Global Values

To create a post-processor whose inputs are only global variables, you can use the global function postprocessor command:

```
POSTPROCESS GLOBAL_FUNCTION "[FUNCTION]" as [OUTPUT]
```

For example:

```
POSTPROCESS AVERAGE OF EXPRESSION TEMPERATURE ON BLOCK_1 AS T_avg
POSTPROCESS AVERAGE OF EXPRESSION DENSITY ON BLOCK_1 AS rho_avg
POSTPROCESS GLOBAL_FUNCTION "T_avg + rho_avg" as T_rho_avg
```

Note: Global functions are evaluated one at a time **in the order specified in the input file**, and are always run last, after reduction and value post-processors.

Global functions may contain circular references. If a global function uses its output value as one of its inputs, the value used will be the output from the prior evaluation. This allows the creation of "latching" functions or state shifts.

For example, to create a variable that switches from 0 to 1 to indicate if the domain ever had an average temperature above 500 at any time in the simulation one could do:

```
POSTPROCESS GLOBAL_FUNCTION                        \$
   "(Tavg>500 && abv500==0) ? 1 : abv500"          \$
   as abv500
```

Alternately, to keep the last 4 time states for a global variable and create a trailing average one could do:

```
POSTPROCESS AVERAGE OF EXPRESSION TEMPERATURE ON BLOCK_1 AS Tavg
POSTPROCESS GLOBAL_FUNCTION "Tavg2" as Tavg3
POSTPROCESS GLOBAL_FUNCTION "Tavg1" as Tavg2
POSTPROCESS GLOBAL_FUNCTION "Tavg0" as Tavg1
POSTPROCESS GLOBAL_FUNCTION "Tavg"  as Tavg0
```

(continues on next page)

(continued from previous page)

```
POSTPROCESS GLOBAL_FUNCTION                \$
    "(Tavg3 + Tavg2 + Tavg1 + Tavg0)/4"   \$
    as T_running_avg
```

On a given step, this will first calculate a new value for `Tavg`, then shift the stored values of `TavgX` back by one position each and finally calculate a trailing average of the last 4 time step values.

Post-Processing Specific Models

By default Aria sums source terms and boundary flux terms together in a single expression (e.g. `ENERGY_SOURCE` or `ENERGY_FLUX`). To post-process specific contributions to these expressions requires an extra argument to select which model you want to post-process.

For example, consider a case where you have two boundary conditions applied and two source terms in an energy equation

```
Source for Energy on block_1 = Melting_Source
Source for Energy on block_1 = Joule_Heating

BC Flux for Energy on surface_1 = Generalized_Rad
BC Flux for Energy on surface_1 = Generalized_Nat_Conv
```

If you wanted to post-process the total energy source and total flux you could use the previously defined commands as:

```
Postprocess value of expression energy_source on block_1 as Esrc
Postprocess value of expression energy_flux on surface_1 as Eflux
```

Note: The name of the expression here (e.g. "energy_flux") comes from the applied BC line. For `BC [TYPE] for [EQ]` the name will be `Postprocess value of expression EQ_TYPE on ...`. If you are using a Calore-style BC block, you must find the equivalent 1-line BC command to determine what name to use here.

However, if you wanted to post-process contributions from individual models you would need to add the model name:

```
Postprocess value of expression energy_source on block_1 \$
  Model Joule_Heating as Esrc_jh

Postprocess value of expression energy_flux on surface_1 \$
  Model generalized_nat_conv as Eflux_conv
```

Universal Expressions

You can define universal expressions to provide pre-calculated quantities for both material models and post-processors. These act as global, spatially varying functions, and can be evaluated anywhere their inputs are available. The `UNIVERSAL ARIA EXPRESSIONS` block is added at the Sierra scope (top-level) and is applied to all Aria regions in the problem.

```
Begin universal aria expressions
  User Expression = [MODEL] user_tag=[NAME] [MODEL_ARGS]
End
```

For example, to define universal expressions for temperature in Celsius (called `tempC`), and radius (called `rad`) you would add:

```
Begin universal aria expressions
  User Expression = Scalar_String_Function          \$
      user_tag = "tempC" f = "temperature - 273.15"

  User Expression = Scalar_String_Function          \$
      user_tag = "rad" f = "sqrt(x^2 + y^2 + z^2)"
End
```

Then you could use this expression in your post-processors like this:

```
Postprocess average of expression tempC on block_1 as TCavg_1
Postprocess average of expression tempC on block_2 as TCavg_2
```

You could also use `tempC` or `rad` as inputs for material properties or in initial conditions and boundary conditions within your Aria regions. For example, to add an initial condition as a function of radius you would use:

```
IC for Temperature on all_blocks = Scalar_String_Function f = "200 +␣
↪30*rad"
```

Postprocessing Existing Results

Occasionally it is useful to postprocess existing results files, for example to convert temperatures from Kelvin to Celsius, or to evaluate an intermediate quantity like thermal conductivity that was not included in the initial results. This is possible in Aria by using a combination of an *IO Region*, *Transfer*, and XFER equations.

The basic steps for postprocessing existing results is to:

1. Copy the input file used to generate the results to a new file.

2. Replace the list of equation terms (MASS, DIFF, etc) with XFER to prevent Aria from re-solving the problem.
3. Comment out or remove all boundary conditions and source terms.
4. Add a *Finite element model* that uses the existing results file as the database.
5. Add an *IO Region* referencing that finite element model, and include the *Timestep adjustment interval = 1* command in that block.
6. Setup a *Transfer* for the solution fields from the existing results file to the Aria region.
7. Add the new desired postprocessors and output specifications to the Aria region. Make sure to set the results output frequency to output every step and change the output file name to not overwrite the existing results.
8. Change the time stepping parameters for the Aria Region to use a fixed time step size with the initial time step size matching the termination time. This will let the IO region control the time step to match the times present in the existing results.
9. Run Aria using the modified copy of the input file.

A complete example of this is presented in *Postprocessing Additional Quantities*.

Note: There are a few limitations to this approach:

1. The existing results file must have all the prerequisite field data to calculate the new quantity that is being postprocessed.
2. Postprocessing of results for problems with Bulk Fluid Elements should use the existing results file for the problem geometry. This will preserve consistency between the element blocks on the grid and results file to facilitate transfer from the IO_Region.
3. Because we switched equations to XFER and commented out boundary conditions and source terms it is not possible to postprocess integrated fluxes on boundaries, applied boundary condition values, or source term values.

Advanced Details

Selecting Post-Processing Domains

To evaluate a post-processor on a single block or surface, you can simply use the block or surface name directly:

```
POSTPROCESS AVERAGE OF EXPRESSION TEMPERATURE ON BLOCK_1 AS T_AVG
```

To evaluate a post-processor on multiple blocks, you can create an assembly or a mesh group (see *Assemblies and Mesh Groups*) and evaluate the post-processor on that group:

```
Mesh Group core_blocks = block_1 block_2 block_3
POSTPROCESS AVERAGE OF EXPRESSION TEMPERATURE ON core_blocks AS T_AVG
```

For combinations of parts involving both part lists and CONTACTING or TOUCHING notation (see below) you may need to use a block-style post-processor, which supports multiple location commands (*Block Style Commands*).

You can also use the *default aliases* ALL_BLOCKS, ALL_SURFACES, and ALL_BULK_ELEMENTS to select all blocks, surfaces, and *bulk nodes* respectively. The following aliases can be used to apply the postprocessor at *each* location

EACH_BLOCK
: Apply the postprocessor on each active element block in the mesh

EACH_SURFACE
: Apply the postprocessor on each active surface in the mesh

EACH_BULK_ELEMENT
: Apply the postprocessor on each active *bulk node* in the mesh

For example, the following command on a mesh containing block_1, block_2, and block_3 will produce four global variables: T_AVG_BLOCK_1, T_AVG_BLOCK_2, T_AVG_BLOCK_3, and T_AVG_TOTAL.

```
POSTPROCESS AVERAGE OF EXPRESSION TEMPERATURE ON EACH_BLOCK AS T_AVG
```

Post-Processing on Contact Surfaces

To postprocess one of the previous quantities on a contact surface, you must use the SURFACE_A CONTACTING SURFACE_B notation for the PART argument. The post-processor will only be evaluated on the portions of SURFACE_A which are in contact with SURFACE_B.

For example:

```
POSTPROCESS VALUE OF EXPRESSION CONTACT_HEAT_TRANSFER_COEFFICIENT \$
    ON SURFACE_1 CONTACTING SURFACE_2 AS CONTACT_HTC
```

Note: Since the postprocessor is run only on the faces of SURFACE_A, discontinuous quantities are NOT averaged. The postprocessed value(s) correspond only to the values on SURFACE_A.

Post-Processing on Surface Subsets

To postprocess one of the previous quantities on a subset of a surface based on what block it is in contact with you can use the SURFACE_A TOUCHING BLOCK_B notation for the PART argument.

The post-processor will only be evaluated on the faces of SURFACE_A which are connected to BLOCK_B.

For example:

```
POSTPROCESS VALUE OF EXPRESSION TEMPERATURE \$
    ON SURFACE_1 TOUCHING BLOCK_2 AS TSURF_2
```

Post-Processing on Internal Sidesets

An internal sideset occurs when a surface exists in between two blocks, as is demonstrated below for some internal surface s shared between block_1 and block_2

```
      top
+------+------+
|      |      |
|  1   s   2  |
|      |      |
+------+------+
```

A postprocessor specified on this internal_surf will contain contributions from the values using block_1s expressions as well as block_2s. If the value is continuous across the interface, this will not affect the result. Special care must be taken when postprocessing a quantity which is discontinuous across an internal surface, or one which is connected to multiple blocks. For value postprocessors, contributions of each block are accounted for as defined by the AVERAGING METHOD specified. For example if an internal surface is shared by two blocks, then

```
POSTPROCESS VALUE OF EXPRESSION DISC_QTY \$
    ON INTERNAL_SURF AS QSURF
```

will contain the contributions of both

```
POSTPROCESS VALUE OF EXPRESSION DISC_QTY \$
    ON INTERNAL_SURF TOUCHING BLOCK_1 AS QSURF_B1
POSTPROCESS VALUE OF EXPRESSION DISC_QTY \$
    ON INTERNAL_SURF TOUCHING BLOCK_2 AS QSURF_B2
```

Similarly for integral reduction types, the average value of the quantity on the surface is taken. For example if an internal surface is exactly shared by two blocks, then

```
POSTPROCESS AVERAGE OF EXPRESSION DISC_QTY \$
    ON INTERNAL_SURF AS AVGQ
```

is equivalent to

```
POSTPROCESS AVERAGE OF EXPRESSION DISC_QTY \$
    ON INTERNAL_SURF TOUCHING BLOCK_1 AS AVGQ_B1
POSTPROCESS AVERAGE OF EXPRESSION DISC_QTY \$
    ON INTERNAL_SURF TOUCHING BLOCK_2 AS AVGQ_B2
POSTPROCESS GLOBAL_FUNCTION \$
    "(AVGQ_B1 + AVGQ_B2)/2" AS AVGQ
```

For nodal operations (Min, Max, and L-Infinity Norm), the postprocessor can be seen as applying the operator over the limit approaching the surface from each block. For example for some internal surface touching 3 blocks:

```
POSTPROCESS MAX OF EXPRESSION DISC_QTY \$
    ON INTERNAL_SURF AS MAXQ
```

is equivalent to

```
POSTPROCESS MAX OF EXPRESSION DISC_QTY \$
    ON INTERNAL_SURF TOUCHING BLOCK_1 AS MAXQ_B1
POSTPROCESS MAX OF EXPRESSION DISC_QTY \$
    ON INTERNAL_SURF TOUCHING BLOCK_2 AS MAXQ_B2
POSTPROCESS MAX OF EXPRESSION DISC_QTY \$
    ON INTERNAL_SURF TOUCHING BLOCK_3 AS MAXQ_B3
POSTPROCESS GLOBAL_FUNCTION \$
    "MAX(MAXQ_B1, MAXQ_B2, MAXQ_B3)" AS MAXQ
```

Note: If a quantity is only defined on one side of an internal sideset, one must specify `TOUCHING` to avoid a missing expression error.

Note: When working with shared surfaces, one must ensure that the mesh is defined such that the sideset is associated with all connected blocks. For example in Cubit, the default behavior of sideset creation connects it to only one block touching the surface. In order for the surface to be `touching` all connected blocks, it must be defined with the `With Respect To` box checked and `wrt volume` set to either `all` or the blocks of interest.

Block Style Commands

All of the prior post-processing line commands can also be written as multi-line block commands if desired. For example, instead of

```
POSTPROCESS VALUE OF EXPRESSION TEMPERATURE ON BLOCK_1 AS TEMP1
```

you could use

```
BEGIN POSTPROCESS VALUE
  SOURCE TYPE = EXPRESSION NAME = TEMPERATURE
  LOCATION = BLOCK_1
  OUTPUT NAME = TEMP1
END
```

The general syntax for the block-style post-processors is

```
BEGIN POSTPROCESS [REDUCTION_TYPE]
  SOURCE TYPE = EXPRESSION NAME = [EXPR]
  LOCATION = [PART]
  OUTPUT NAME = [OUTPUT]
  [WEIGHT FUNCTION = "WFCN"]
  [MODEL = MODEL_NAME]
END

BEGIN POSTPROCESS [REDUCTION_TYPE]
  SOURCE TYPE = FUNCTION "[FUNCTION]"
  LOCATION = [PART]
  OUTPUT NAME = [OUTPUT]
END
```

For reference, the syntax for the block style commands is outlined in *Postprocessing*. The primary advantage of the block style command is that you can use multiple `LOCATION` commands to evaluate the post-processor on multiple parts without using a mesh group or assembly. This allows more complicated combinations of conditions that would not be possible with the line-style command. For example:

```
BEGIN POSTPROCESS AVERAGE
  SOURCE TYPE = EXPRESSION NAME = TEMPERATURE
  LOCATION = SURFACE_1
  LOCATION = SURFACE_3 TOUCHING BLOCK_2
  LOCATION = SURFACE_4 CONTACTING SURFACE_5
  OUTPUT NAME = TAVG_SURFS
END
```

Self Referencing Field Postprocessing

Similar to *Global Values*, self-referencing string functions are supported in value postprocessors. This allows running fields throughout a simulation. E.g. to get the maximum temperature observed at each node over a simulation, one could use the following

```
POSTPROCESS VALUE OF FUNCTION "TEMPERATURE > MAX_TEMP ? TEMPERATURE : MAX_
↪TEMP" \$
  on block_1 as MAX_TEMP
```

Note that the initial value taken for a self-referenced field is 0 (e.g. `MAX_TEMP` in this example). To start with a different initial value, one can declare the initial value as a user field e.g.

```
USER FIELD REAL NODE SCALAR MIN_TEMP ON block_1 VALUE = 1e10
POSTPROCESS VALUE OF FUNCTION "TEMPERATURE < MIN_TEMP ? TEMPERATURE : MIN_
↪TEMP" \$
  on block_1 as MIN_TEMP
```

Binary State Fields

In some cases, a user may wish to track a "state" condition that preserves history as time advances. An example is a material that undergoes an irreversible phase change once it reaches a certain temperature. In this scenario, there is a need to track all nodes or elements that have ever gone above the phase change temperature, not just entities that are above the temperature at the current time step, so that material properties can be set accordingly. This could be accomplished with self-referencing string functions like the following for a nodal field

```
POSTPROCESS VALUE OF FUNCTION "TEMPERATURE > 1500 ? 1.0 : PHASE_CHANGED" \
↪$
  on block_1 as PHASE_CHANGED
```

Or like this for an element field

```
POSTPROCESS VALUE OF FUNCTION "TEMPERATURE > 1500 ? 1.0 : PHASE_CHANGED" \
↪$
  on block_1 as PHASE_CHANGED using P0
```

However, Aria also provides a shorthand for these operations called binary state fields. Binary state fields are intended to represent a spatially varying two state condition that can be toggled on or off when specified criteria are met but otherwise maintains its state as time advances. Binary state fields will also persist through restarts, which may not always be the case for self-referencing string functions. The common use case for binary state fields is controlling material properties, so they are specified in the material block as opposed to the Aria region or equation system blocks like other postprocessors.

```
Begin Aria Material material1
  User Expression = Binary_State User_Tag = my_binary_state Initial_State␣
↪= Inactive \$
    Activation_Function = "element_average_temperature > 1000" Location =␣
↪Element
End
```

The above example would create a binary state field that tracks elements whose average temperature has gone above 1000. Aria will automatically generate the necessary postprocessors to populate the defined binary state fields on any of the element blocks the material is used on. Binary state fields cannot be created on surfaces or edges. These postprocessors will be added to the last defined equation system, meaning that binary state fields will always be updated at the very end of a time step. Expressions using a binary state field during an equation solve will use the previous time step's value. The active state of a binary state field is defined as 1.0 and the inactive as 0.0. This allows them to be easily used in string function conditionals such as the below example that switches thermal conductivity between two values depending on the state

```
Thermal Conductivity = Scalar_String_Function f = "my_binary_state ? 0.1␣
↪: 10"
```

Element binary state fields can also make use of element averaged expressions. These can be specified in the activation or deactivation criteria using the syntax `ELEMENT_AVERAGE_[EXPRESSION]`. Element average expression are the volume average of the specified expression over the element.

Nodal Averaging Method

Depending on their continuity, expressions may not have unique values at the nodes of the mesh. For example, any expression that is derived from a gradient operation of a Q1 interpolation can be multi-valued at a given node (e.g., darcy velocity derived from Q1 pressure field, heat flux vector derived from Q1 temperature field, etc.).

Aria supports three nodal averaging methods (i.e., `SUPPORT_WEIGHTED`, `SUPPORT_AVERAGE`, `SIMPLE`) that can be applied in these situations to output a unique post-processed value at the nodes of the mesh using the following line command

```
POSTPROCESS VALUE OF EXPRESSION MYEXPR ON BLOCK_1 \$
    AS MY_AVERAGED_NODAL_FIELD AVERAGING METHOD = [METHOD]
```

In most cases, the SUPPORT_WEIGHTED method (DEFAULT) is a good choice and is defined as

$$f_i = \frac{\sum_{e \in E} f_e(\boldsymbol{x}_i) \int_{\mathrm{V}_e} \phi_e^i(\boldsymbol{x})\, w_e(\boldsymbol{x})\, \mathrm{dV}}{\sum_{e \in E} \int_{\mathrm{V}_e} \phi_e^i(\boldsymbol{x})\, w_e(\boldsymbol{x})\, \mathrm{dV}}$$

where E is the set of elements e connected to node i, $\phi_e^i(\boldsymbol{x})$ is the element shape function for node i and where $f_e(\boldsymbol{x}_i)$ is the expression value evaluated at node i using the local element shape functions. Note $f_e(\boldsymbol{x}_i)$ may be multi-valued at node i, hence the subscript e. Lastly, $w_e(\boldsymbol{x})$ is an optional weighting that can be applied via the WITH WEIGHTING syntax e.g.,

```
POSTPROCESS VALUE OF EXPRESSION MYEXPR ON BLOCK_1 \$
    WITH WEIGHTING "DENSITY" AS MY_MASS_AVERAGED_NODAL_FIELD AVERAGING␣
↪METHOD = SUPPORT_WEIGHTED
```

When WITH WEIGHTING is not specified, one can interpret the SUPPORT_WEIGHTED nodal value as a volume weighted average i.e.,

$$f_i = \frac{\sum_{e \in E} f_e(\boldsymbol{x}_i) \Delta V_e^i}{V_i}$$

where $\Delta V_e^i = \int_{\mathrm{V}_e} \phi_e^i(\boldsymbol{x})\ \mathrm{dV}$ is the volume contribution to node i from connected element e and where $V_i = \sum_{e \in E} \Delta V_e^i$ is the total nodal volume for node i. As an aside, note that the finite element shape functions form a partition of unity and so the volume of an element will always be completely accounted for through the sum of its contributions to it nodes.

Similarly, the SUPPORT_AVERAGED method is defined as

$$f_i = \frac{\sum_{e \in E} \int_{\mathrm{V}_e} f_e(\boldsymbol{x}) \phi_e^i(\boldsymbol{x})\, w_e(\boldsymbol{x})\, \mathrm{dV}}{\sum_{e \in E} \int_{\mathrm{V}_e} \phi_e^i(\boldsymbol{x})\, w_e(\boldsymbol{x})\, \mathrm{dV}}$$

where $f_e(\boldsymbol{x})$ is now the expression value at a point within the element. As such, this nodal averaging method corresponds to an average of the expression within the support domain.

Lastly, the SIMPLE method is defined as the numeric average of the nodal values:

$$f_i = \frac{\sum_{e \in E} f_e(\boldsymbol{x}_i)}{N_E}$$

Note: If an expression corresponds to a field interpolation (i.e., COEFF_OP) then the nodal averaging operations are skipped and the known unique nodal value is simply returned.

Legacy Postprocessor Conversion

The suite of postprocessors described above are an improvement on the legacy postprocessing capabilities that preceded it, adding additional functionality, concise syntax, and GPU support. As part of the migration to this suite, a conversion tool is provided. The tool is available with a `sierra` module, e.g.

```
$ module load sierra
$ legacyPP2aria legacy_file.i native_aria_file.i
```

and a detailed explanation of its syntax can be found in the help description

```
$ legacyPP2aria --help
```

This tool takes a given input file, and converts all legacy postprocessors to the corresponding native Aria postprocessors in the given output file. This typically has the added benefit of making the input more concise. For example

```
begin field function tempK
  Use Nodal Field Solution->Temperature
end

begin average value postprocessor avg_temp_block_1
  use function tempK
  Volumes block_1
end

begin min max postprocessor max_temp_block_1
  use function tempK
  Volumes block_1
  compute max
end

begin evaluate function postprocessor pt1_temp_block_1
  At Nearest Node
  Location -0.015 0 -0.02
  Volumes block_1
  Use Function tempK
```

(continues on next page)

(continued from previous page)

```
end

...

begin aria region myRegion
  ...

  evaluate postprocessor avg_temp_block_1
  evaluate postprocessor max_temp_block_1
  evaluate postprocessor pt1_temp_block_1
end
```

becomes

```
begin aria region myRegion
  ...

  postprocess average of expression temperature on block_1 as avg_temp_
↪block_1
  postprocess max of expression temperature on block_1 as max_temp_block_1
  postprocess value of expression temperature at point -0.015 0 -0.02 as␣
↪pt1_temp_block_1
end
```

If the script fails to convert a line, it will keep the original postprocessor intact and will prepend a comment stating why it failed e.g.

```
# Unable to convert post processor. [REASON]
begin average value postprocessor foo
  ...
end
```

Note: The `legacyPP2aria` tool is not meant to cover all corner cases of use. If you are stuck, reach out to sierra-help@sandia.gov for help converting legacy postprocessors to native Aria equivalents.

4.7 Solving the problem

Nonlinear Solution Strategy

Equation System Overview

In coupled physics problems one is tasked with solving a set of nonlinear coupled equations and ideally one would solve these equations as a fully-coupled monolithic system. In practice solving the set of equations monolithically may prove impractical owing to large linear system memory requirements and difficulty in locating a small radius of convergence. Thus in some cases one may wish to perform a segregated solve of the equations where each set of equations can be solved in a slightly different manner. This ability to solve subsets of the entire equation set is supported through a feature called Equation System by which criteria for solving a set of nonlinear equations can be applied to a select group of equations.

By default a single set of coupled equations is treated internally as an Equation System without need of additional input syntax. Here the solution strategy for the Equation System is solely determined by a collection of *nonlinear solution specifications*. A typical setup for a single equation system might be

```
Begin Aria Region myRegion
  .
  Use Linear Solver solve_temperature
  .
  eq energy for temperature On block_1 using q1 with lumped_mass diff
  .
  nonlinear solution strategy = newton
  use dof averaged nonlinear residual
  maximum nonlinear iterations = 20
  nonlinear residual tolerance = 1.e-6
  .
  .
End Aria Region myRegion
```

Simulations in which different subsets of the coupled equations are solved in a segregated manner require that each subset be explicitly defined as an Equation System with a corresponding solution strategy which may differ for each equation system. When multiple Equation Systems are present, the overall segregated solution is governed by segregated iteration and convergence criteria at an *outer level*.

A typical setup for segregated solution of multiple equation systems might be

```
Begin Aria Region myRegion
  .
```

(continues on next page)

(continued from previous page)

```
  Segregated global nonlinear residual tolerance = 6.0e-9
  .
  Begin Equation System temperature
    .
    eq energy for temperature On block_1 using q1 with lumped_mass diff
    Use Linear Solver solve_temperature
    .
    nonlinear solution strategy = newton
    use dof averaged nonlinear residual
    maximum nonlinear iterations = 20
    nonlinear residual tolerance = 1.e-6
    .
  End Equation System temperature
  .
  Begin Equation System species
    .
    eq species for species_0 On block_1 using q1 with diff
    .
    Use Linear Solver solve_species
    .
    nonlinear solution strategy = newton
    use dof averaged nonlinear residual
    maximum nonlinear iterations = 10
    nonlinear residual tolerance = 1.e-8
    .
  End Equation System species
  .
End Aria Region myRegion
```

Nonlinear Solution Specifications

The general solution strategy used in Aria is to solve the requested system of equations for solution variables $\boldsymbol{\phi}$ incrementally. In the context of a Newton formulation then within the Newton step one will solve

$$\boldsymbol{J}\Delta\boldsymbol{\phi} = -R(\boldsymbol{\phi})$$

where $\boldsymbol{J}$ is the Jacobian matrix, $\Delta\boldsymbol{\phi}$ is the solution increment and $R(\boldsymbol{\phi})$ is the system residual for the *governing equations*. See the *theory section* for more information on the solution strategy. This chapter describes basic nonlinear solution command line settings that influence how a converged solution will be obtained. For transient solutions these settings are used in combination with *time integration settings* for advancing a solution to the next time step.

It is instructive here to review a typical log file output of the Aria solution summary in order to establish how a user might influence the solution procedure.

```
Equation System AriaRegion->main:
* Step  : Transient, Strategy: NEWTON, Time: 2.20e+03, Step: 5.29e+00
* Matrix: Solver: "Solve_Temperature", Unknowns: 1360, Nonzeros: 11578
* Mesh  : Processor 0 of 1: 1253 of 1253 elems, 1363 of 1363 nodes
* Computing View Factors for enclosure space
  N O N L I N E A R                L I N E A R
----------------------   -----------------------------------
Step  Resid    Delta     Itns Status  Resid    Asm/Slv Time
---- -------- --------   ---- ------ -------- ---------------
1    6.37e-02 1.10e+01   59    ok    8.23e-07 7.0e-03/2.0e-03
2    1.10e-04 8.18e-03   58    ok    1.19e-09 7.0e-03/2.0e-03
3    3.06e-07 8.81e-05   61    ok    3.68e-12 6.0e-03/2.0e-03
Termination reason: 8.80515e-05 < nonlinear_correction_tolerance(0.001)
```

In the log file output shown above, columns under the `N O N L I N E A R`, heading are related to the command lines described in this chapter. The columns `Resid` and `Delta` represent L2 norms of an aggregate residual $R^*(\boldsymbol{\phi})$ and solution increment vector $\Delta\boldsymbol{\phi}$ while the column denoted `Step` denotes the iteration within this Newton step. Columns under the `L I N E A R` heading represent the number of linear solver iterations (`Itns`), the `Resid` column represents the linear system residual $R^*(\Delta\boldsymbol{\phi})$. Here $R^*(\Delta\boldsymbol{\phi}) = \boldsymbol{J}\Delta\boldsymbol{\phi} + R(\boldsymbol{\phi})$, the residual for the assembled system of equations, is solved for $\Delta\boldsymbol{\phi}$ to the tolerance requested in the Linear Solver specifications. The column labeled `Status` indicates whether the linear solve completed successfully, `ok` or `fail`, within the allowable number of iterations defined in the Linear Solver specifications. The column labeled `Asm/Slv Time` contain assembly and solution times for the system of equations.

Given a current value of $\boldsymbol{\phi}$ then within each Newton step the residual $R^*(\Delta\boldsymbol{\phi})$ is first evaluated using $\Delta\boldsymbol{\phi} = 0$ and tested for convergence. Here the convergence conditions are satisfied when either the `NONLINEAR RESIDUAL TOLERANCE` or `NONLINEAR CORRECTION TOLERANCE` are achieved. If the solution is not yet converged, the system of equations is turned over to the equation solver where it is solved for a new $\Delta\boldsymbol{\phi}$ subject to the Linear Solver specifications. Then the Solver solution is used to update the nonlinear solution $\boldsymbol{\phi} = \boldsymbol{\phi}_{old} + \Delta\boldsymbol{\phi}$ and the process begins anew subject to a user supplied `MAXIMUM NONLINEAR ITERATIONS`.

Within the Newton step a solution is said to `FAIL` if it is not converged within `MAXIMUM NONLINEAR ITERATIONS` and this occurrence will be noted in the log file. For transient simulations one will normally repeat the solution procedure with a smaller time step as described in *Time Step Selection*. In some cases the user may chose to override the `FAIL` condition using the `ACCEPT SOLUTION AFTER MAXIMUM NONLINEAR ITERATIONS` command line but *one is cautioned that this option should be used with discretion as it may discredit any expectation of accuracy in transient results*. For completeness, a summary of pass, fail, and fail but accepted steps are included at the end of the log file.

For most applications the default `NONLINEAR RESIDUAL TOLERANCE` (1e-6) is adequate when

accompanied by a Linear Solver specification of `RESIDUAL NORM TOLERANCE` minimum of 1.0e-8. Inclusion of the command line `USE DOF AVERAGED NONLINEAR RESIDUAL` is highly recommended, particularly when performing uniform mesh refinement studies. A summary of these and other nonlinear solution related command lines is given in what follows.

Linear Solution Strategy

Overview

Aria supports several choices of linear solver packages. Starting with the 5.0 release the Tpetra based linear solvers are the only linear solvers supported.

Within the input file linear solver command block will appear at the Domain scope as illustrated below. Note that different linear solvers can be used for different equation systems. Furthermore, a solver command block can appear in the input but not be used.

```
Begin Sierra
  .
  .
  Begin Tpetra Equation Solver third_solver
    .
    Tpetra linear solver commands
    .
  End
  .
  Begin Procedure My_Aria_Procedure
     .
     Begin Aria Region My_Region
       .
       use linear solver first_solver
       .
     End

     Begin Aria Region My_Other_Region
       .
       use linear solver third_solver
       .
     End

  End
  .
End Sierra
```

See *Solver Selection Guidelines* for suggestions on how to select a type of linear solver for

different kinds of problems Aria can solve.

Tpetra Solver Overview

The syntax for specifying a Tpetra-based solver uses a nested command block structure. The outermost level at the Sierra scope is the **BEGIN** `TPETRA EQUATION SOLVER` block. This block must contain a single nested solver block corresponding to your desired solver type. Several types of direct and iterative linear solvers are provided through the Amesos2 and Belos packages of Trilinos. The supported solver blocks are listed in *Solvers*. If one of the iterative solvers is selected, it must also have a nested preconditioner block. A variety of preconditioners are supported through the Ifpack2 and MueLu Trilinos packages. See *Preconditioners* for details on the supported preconditioners and their available options.

Solver Error Codes

If the linear solve fails it will report an error code. While these codes are not universally standardized across linear solvers, they can generally be interpreted with the following rules:

Error 1 (or -1)
: The linear solver reached the maximum number of linear solver iterations and did not reach a solution. Check the final linear solver residual to see if this is serious or not. Even if the error is not serious (residuals are small) this usually means the solve is much more expensive than necessary and a better solver/preconditioner combination should be used.

Error 2 (or -2, 4, or -4)
: The linear solver encountered a fatal internal error. This is often because of a singular matrix or a NaN or Infinite term. This is almost always a serious error.

Error 3 (or -3)
: The linear solver had an internal loss of precision. This is often not a serious error, but may indicate the need for a better solver/preconditioner combination. As always, check the final linear solver residual reported in the log file to determine if it is serious or not.

Expert Syntax

The standard solver and preconditioner blocks only support a subset of the commonly used options that are supported by the underlying Trilinos packages. The subset was chosen to cover the vast majority of linear solver use cases within Aria simulations, but an alternative approach is provided for expert users who would like to be able to pass parameters directly to the underlying Belos, Ifpack2, and MueLu packages. This expert interface is used by providing the path to an XML file containing the definition for a `Teuchos::ParameterList` that is supported by the

underlying Trilinos package in one of the **BEGIN** EXPERT BELOS SOLVER, **BEGIN** EXPERT IFPACK2 PRECONDITIONER, or **BEGIN** MUELU PRECONDITIONER blocks as desired.

Sample XML for expert Belos solver

```
<ParameterList>
  <Parameter name="solution method" type="string" value="CG"/>
  <ParameterList name="Belos-Tpetra">
    <Parameter name="Convergence Tolerance" type="double" value="1.e-12"/>
    <Parameter name="Maximum Iterations" type="int" value="1000"/>
  </ParameterList>
</ParameterList>
```

The `solution method` parameter is required. Any additional parameters that are not specified in the `Belos-Tpetra` sublist will use their default values as defined by Belos.

Sample XML for expert Ifpack2 preconditioner

```
<ParameterList>
  <Parameter name="preconditioner type:" type="string" value="ILUT"/>
  <Parameter name="fact: ilut level-of-fill" type="double" value="1.0"/>
  <Parameter name="fact: drop tolerance" type="double" value="0.0"/>
</ParameterList>
```

The `preconditioner type:` parameter is required. Any additional parameters that are not specified will use their default values as defined by Ifpack2.

Sample XML for expert MueLu preconditioner

```
<ParameterList name="MueLu-Tpetra">
  <Parameter        name="verbosity"                        type="string"␣
↪  value="high"/>
  <Parameter        name="coarse: type"                     type="string"␣
↪  value="ILUT"/>
  <Parameter        name="coarse: max size"                 type="int"   ␣
↪  value="1000"/>
  <Parameter        name="max levels"                       type="int"   ␣
↪  value="4"/>
  <Parameter        name="smoother: type"                   type="string"␣
↪  value="CHEBYSHEV"/>
  <ParameterList    name="smoother: params">
     <Parameter name="chebyshev: degree"                    type="int"   ␣
↪  value="2"/>
     <Parameter name="chebyshev: ratio eigenvalue"          type="double"␣
↪  value="20"/>
     <Parameter name="chebyshev: min eigenvalue"            type="double"␣
```

(continues on next page)

(continued from previous page)

```
↪  value="1.0"/>
    <Parameter name="chebyshev: zero starting solution"    type="bool"  ␣
↪  value="true"/>
    <Parameter name="chebyshev: eigenvalue max iterations" type="int"   ␣
↪  value="15"/>
  </ParameterList>
  <Parameter        name="aggregation: type"               type="string"␣
↪  value="uncoupled"/>
  <Parameter        name="aggregation: drop tol"           type="double"␣
↪  value="0.02"/>

  <Parameter        name="repartition: enable"             type="bool"  ␣
↪  value="true"/>
  <Parameter        name="repartition: min rows per proc"  type="int"   ␣
↪  value="1000"/>
  <Parameter        name="repartition: start level"        type="int"   ␣
↪  value="2"/>
  <Parameter        name="repartition: max imbalance"      type="double"␣
↪  value="1.327"/>
  <Parameter        name="repartition: partitioner"        type="string"␣
↪  value="zoltan2"/>
</ParameterList>
```

Unspecified parameters will use their default values as defined by MueLu.

For further documentation of the parameters supported by each Trilinos package please see the corresponding documentation on trilinos.org.

Solver Selection Guidelines

Selecting an appropriate linear solver that is both robust and efficient for a particular Aria simulation can be confusing. In general there is a tradeoff between solver speed and robustness, and this can significantly affect the total time to solution. The following guidelines for choosing a linear solver may be useful:

1. As a starting point preset solvers are provided for several common types of problems. These are recommended as a starting point for any analysis.

- The `teko_multiphysics` preset solver type is recommended as a good starting point for a wide range of problems. It uses a GMRES solver with block multiphysics preconditioning based on the heuristic algorithms described in *Teko Heuristic Multiphysics Preconditioner*. For single physics use-cases, this employs an adaptive solver approach that monitors the convergence of the linear solver. The adaptive solver starts with a GMRES solver preconditioned via Jacobi (similar to the `thermal` preset solver) and progressively switches

to more robust DD-ILU preconditioners (similar to the `multiphysics` preset solver) as needed. Note the particular details of this adaptive solver described in *Teko Heuristic Multiphysics Preconditioner* are subject to change, but will reflect the 'latest and greatest' solver settings.

- The `thermal` preset solver type is recommended as the starting point for general thermal analysis problems. It uses a GMRES solver with Jacobi preconditioning and is a good balance between runtime per linear solver iteration and robustness for most thermal problems.
- The `thermal_symmetric` and `thermal_bicgstab` presets can be slightly faster for thermal analysis problems that do not involve contact. They are CG and BiCGSTAB solvers respectively, each with Jacobi preconditioning. These trade some robustness for shorter runtime per linear iteration when compared with the GMRES solver used by the `thermal` preset.
- The `thermal_multigrid` preset is found to be effective for general scalar equation analysis, including thermal problems and for scalar problems that often require direct solves.
- For mesh displacement and elasticity problems, the `displacement_multigrid` preset is recommended.
- For fluids problems the `continuity` preset is recommended for the continuity equation system, and the `scalar_transport` preset is recommended for the momentum equation system, as well as any additional scalar advection-diffusion equations. The `continuity` preset is a GMRES solver with an algebraic multigrid preconditioner provided through the MueLu Trilinos package. The `scalar_transport` preset is a GMRES solver with a SGS preconditioner.
- For more general multiphysics problems involving multiple coupled equations in a single equation system the `multiphysics` or `teko_multiphysics` preset solver is the recommended starting point. The former is a GMRES solver with a DD-ILU preconditioner. The latter is a GMRES solver with a heuristically determined Teko preconditioner.
- When using any of the preset solver types, an equivalent solver block is written to the log file and can be used as a starting point for modifying additional solver or preconditioner settings. This solver block will contain the preset preconditioner along with default solver settings. In reusing the this block, users will often choose to increase the number of `MAXIMUM ITERATIONS` in hopes of lowering the solution time, but it is noted that increases beyond triple the default setting reap low reward in obtaining a solution.

2. If the suggested preset solver shows evidence of struggling to solve a particular problem in the log file (for example frequently approaching or reaching the maximum number of linear iterations) consider using a stronger solver and/or preconditioner. For the solvers move from CG to BiCGSTAB to GMRES. If already using GMRES try increasing the number of `RESTART ITERATIONS` to match the `MAXIMUM ITERATIONS`. For the preconditioners move from Jacobi to SGS to DD-ILU or DD-ILUT. Initially leave all preconditioner-specific options to their default values.

3. If GMRES/DD-ILU still shows evidence of struggling to solve the problem consider increasing the ILU fill and overlap levels. First trying overlap of 1, then fill of 1, then both is a good balance between the cost and robustness of the preconditioner. See *Preconditioners* for setting the fill and overlap.
4. If GMRES/DD-ILU with fill and overlap is still insufficient consider:

- Trying one of the direct solvers (`KLU2`, `SuperLU`, or `UMFPACK`) if your problem is small. These are the most robust option available to Aria, but significantly more expensive in both memory usage and runtime than the iterative solver options. They also do not scale in parallel effectively, and should not be used for problems requiring more than a single compute node on HPC systems.
- For multiphysics problems involving many equations in a single monolithic equation system consider moving to a segregated solution scheme involving multiple smaller equation systems. This will make the linear solves easier, but can make nonlinear convergence of the overall problem more challenging. Loosely coupled equations are more likely to be successfully split into separate equation systems without causing nonlinear convergence problems.
- For multiphysics problems where splitting the monolithic equation system into separate equation systems is unfeasible, block preconditioning strategies through Teko may prove effective. This provides extensive flexibility in constructing physics-based preconditioners tailored to the multiphysics problem. As a consequence, setting up block preconditioners is challenging. Fortunately, the `teko_multiphysics` preset solver can automatically generate a Teko preconditioner based on the methods described in *Teko Heuristic Multiphysics Preconditioner*. If you need to manually construct a Teko preconditioner, consult *Block Preconditioning* for additional information and tips on setting up block preconditioners for multiphysics problems.

5. Multigrid preconditioning can be extremely effective and exhibit excellent parallel scalability, however setup of multigrid preconditioners is an extremely complex topic that is both problem type and problem size dependent. Aria supports general algebraic multigrid preconditioning through the MueLu package from Trilinos, and can read MueLu XML files for defining the multigrid setup. Please consult the MueLu documentation for suggestions on selecting a multigrid preconditioner. The default `MUELU` preconditioner is not recommended for general use. However, it is sometimes useful for steady state scalar equation simulations, Poisson problems that are nearly singular and difficult to solve with preset solvers.

Linear Solvers on the GPU

Performant linear solvers for GPUs in general are an open research question. This section contains some guidelines for users in selecting a linear solver (and more importantly preconditioner) when using Aria on GPU platforms.

- All solvers and preconditioners are expected to run correctly when Aria is compiled for GPU platforms, however not all are able to use the GPU and therefore will have poor performance.
- All of the iterative solvers (CG, BiCGSTAB, and GMRES) can run with good performance on GPUs.
- Jacobi preconditioning has excellent GPU performance when it is an effective preconditioner.
- The DD-ILU preconditioner is capable of running on the GPU, but performance can be limited by inherent data dependencies in the ILU factorization and solve algorithms. By default Aria will use an alternative matrix reordering algorithm (metis) when run on the GPU that reduces those data dependencies, however it typically leads to slightly higher linear iteration counts than the RCM reordering method that is the CPU default. Despite increased iteration counts, users can typically expect comparable or slightly improved linear solver runtimes for DD-ILU when compared to a similar CPU machine.
- The SGS preconditioner is an inherently serial algorithm that does not run on the GPU. There is an SGS2 preconditioner that uses a modified algorithm to expose parallelism to run on the GPU that users can try, but increasing the number of linear solver iterations and using Jacobi may result in a lower time to solution.
- The DD-ILUT preconditioner is also an inherently serial algorithm that does not run on the GPU.
- There are experimental FastILU and DD-PARILUT preconditioners available that use iterative approximations to the classical DD-ILU and DD-ILUT algorithms to expose parallelism for improved GPU performance. The development team is still evaluating these preconditioner options and they are not currently recommended for production use by users.
- Teko block preconditioners can be efficient on the GPU, provided each of the sub-block solvers is also performant on the GPU. For example, an iterative sub-block solver with Jacobi preconditioning outperforms DD-ILU, provided the iterative solver converges in sufficiently few iterations.

Tolerance Settings

In addition to selecting the solver and preconditioner type, selecting appropriate convergence tolerance and scaling settings is also important. The default settings (R0 scaling and 1e-6 tolerance) are a reasonable starting place, but may need to be adjusted depending on the problem and nonlinear solver settings. Typically setting the `RESIDUAL SCALING` to either `NONE` or `R0` is most useful.

- `NONE` applies no scaling to the residual for determining linear solver convergence. If the problem has a low residual to being with the solver will only take a few iterations, however if the initial residual is large this option may take more linear solver iterations than is ideal. The linear solver tolerance selected with this option should be at least an order of magnitude lower than the nonlinear residual tolerance used in the Newton solve or the nonlinear problem may have trouble converging.
- `R0` scales the residual by the initial residual. This is useful if you want a looser acceptance tolerance when the problem starts with high residual, however if the problem starts with a small residual this may cause the solver to take an unreasonably large number of iterations. If the initial residual is large and the linear solver tolerance is too loose this can result in the linear solve producing bad/poorly solved results which may cause convergence problems.

An autonomous linear convergence tolerance can be computed within Aria, and is declared within the Tpetra solvers by setting the parameter `CONVERGENCE TOLERANCE = AUTOMATIC` in the Tpetra solver block. This option computes a linear convergence tolerance for each nonlinear iteration based on the initial linear residual and the user-specified nonlinear residual tolerance. The user can also specify an optional minimum linear residual tolerance (with a default value of 1×10^{-6}) to ensure that the computed linear convergence tolerance within Aria is not used if it's too small. Below is an example of using this autonomous setting with a specified minimum

```
BEGIN TPETRA EQUATION SOLVER  GMRES_DDILU
  BEGIN GMRES SOLVER
    BEGIN DD-ILU PRECONDITIONER
    END
    MAXIMUM ITERATIONS = 1000
    RESIDUAL SCALING = NONE
    CONVERGENCE TOLERANCE = AUTOMATIC MINIMUM = 1.0E-8
  END
END TPETRA EQUATION SOLVER
```

Block Preconditioning

The advice contained in this section is relevant to users trying to *manually* construct a Teko-based preconditioner. The `teko_multiphysics` preset solver provides an automated method of generating a Teko-based preconditioner based on the methods described in *Teko Heuristic Multiphysics Preconditioner*.

Overview

Aria supports block preconditioning through the Teko package [33]. Teko support in Aria is currently an expert user feature. Aria currently offers two modes of interacting with Teko. The first is provided through the `TEKO PRECONDITIONER` block, which interfaces directly with the `TPETRA EQUATION SOLVER` blocks in an Aria input deck. The second, provided through the `EXPERT TEKO PRECONDITIONER` block, offers users a way to use Teko is through the expert interface that uses an XML file specify the relevant Teko parameters. Please see https://trilinos.github.io/teko.html or *Expert Teko Syntax* for more information on how to structure this XML.

Note: The `TEKO PRECONDITIONER` and `EXPERT TEKO PRECONDITIONER` blocks currently only support using the `GMRES SOLVER` block.

Block Gauss-Seidel

Teko provides the capability to construct block-based preconditioners using the Block Gauss-Seidel method. For many multiphysics problems, this is a reasonable first choice. Aria requires the mapping from DOFs to sub-block in the Teko preconditioner. An example input file block is shown below.

```
BEGIN TPETRA EQUATION SOLVER GMRES_TEKO_BGS
  BEGIN GMRES SOLVER
    BEGIN TEKO PRECONDITIONER
      DEFINE MATRIX SUBBLOCK NUMBER=1 DOF=VELOCITY
      DEFINE MATRIX SUBBLOCK NUMBER=2 DOF=PRESSURE

      BEGIN TEKO SUBBLOCK PRECONDITIONER DDILU_PC
        BEGIN DD-ILU PRECONDITIONER
        END
      END

      BEGIN TEKO SUBBLOCK PRECONDITIONER AMG_PC
```

(continues on next page)

(continued from previous page)

```
        BEGIN MUELU PRECONDITIONER
        END
      END

      BEGIN BLOCK GAUSS-SEIDEL SOLVER
        USE INVERSE DDILU_PC FOR SUBBLOCK 1
        USE INVERSE AMG_PC FOR SUBBLOCK 2
      END
    END
  END
END TPETRA EQUATION SOLVER
```

The Teko input block starts by defining the DOF-to-block mapping through the `DEFINE MATRIX SUBBLOCK` command. Note that the DOF to sub-block relationship need not be one-to-one, as many DOFs can be mapped to the same sub-block. This gives the user tremendous flexibility in customizing the block preconditioner. For additional guidance on determining good DOF to sub-block mappings, refer to the advice in *Road-map for Constructing Teko Solvers*. Once the DOF-to-block mapping is specified, the means to invert the block system and each sub-block are required. These are both specified through the block solver specification inside the `TEKO PRECONDITIONER` scope. Currently, `BLOCK GAUSS-SEIDEL`, `BLOCK JACOBI`, and `SIMPLE` are supported. Each of these block solvers provide a `USE INVERSE <METHOD> FOR SUBBLOCK <N>` command to describe how to solve each sub-block, where the method name refers to a solver or preconditioner defined in a separate `TEKO SUBBLOCK SOLVER` or `TEKO SUBBLOCK PRECONDITIONER` block.

In the example above , `BLOCK GAUSS-SEIDEL` is the main preconditioning method. Sub-block 1 (velocity) uses a DD-ILU preconditioner to solve the sub-block. Sub-block 2 (pressure), on the other hand, uses MueLu to construct an algebraic multigrid preconditioner to solve the sub-block. The Teko preconditioning feature in Aria allows the specification of sub-block solvers and preconditioners through the `TEKO SUBBLOCK SOLVER` and `TEKO SUBBLOCK PRECONDITIONER` block commands, respectively. Each resembles the syntax of the `TPETRA EQUATION SOLVER` blocks used to solve linear systems in Aria. For advice on choosing appropriate sub-block preconditioners, refer to *Road-map for Constructing Teko Solvers*.

Navier–Stokes and SIMPLE

Note: When using SIMPLE for Navier–Stokes, velocity must be assigned to block 1 and pressure to block 2.

While the Block Gauss-Seidel preconditioners provided by Teko are effective for preconditioning multiphysics problems, they tend to be sub-optimal for Navier–Stokes. Teko provides several

block preconditioning strategies specifically for the solution of the incompressible Navier–Stokes equations. One such method is inspired by the Semi-Implicit Method for Pressure Linked Equations (SIMPLE) and SIMPLEC operating splitting techniques. For a more complete description of this approximate block approximation, refer to [45]. An example input file block for setting up a SIMPLEC block preconditioned is shown below. Extending our previous example input file block from the section on *Block Gauss-Seidel*, we have the following input:

```
BEGIN TPETRA EQUATION SOLVER GMRES_TEKO_SIMPLE
  BEGIN GMRES SOLVER
    BEGIN TEKO PRECONDITIONER
      # NOTE: velocity must be assigned to block 1 for SIMPLE
      DEFINE MATRIX SUBBLOCK NUMBER=1 DOF=VELOCITY
      DEFINE MATRIX SUBBLOCK NUMBER=2 DOF=PRESSURE

      BEGIN TEKO SUBBLOCK PRECONDITIONER DDILU_PC
        BEGIN DD-ILU PRECONDITIONER
        END
      END

      BEGIN TEKO SUBBLOCK PRECONDITIONER AMG_PC
        BEGIN MUELU PRECONDITIONER
        END
      END

      BEGIN SIMPLE SOLVER
        USE INVERSE DDILU_PC FOR SUBBLOCK 1
        USE INVERSE AMG_PC FOR SUBBLOCK 2
        EXPLICIT VELOCITY INVERSE TYPE = ABSROWSUM # or LUMPED, DIAGONAL
        ALPHA = 0.9 # Pressure Schur-Complement (under/over) relaxation, >
↪ 0.0
      END
    END
  END
END TPETRA EQUATION SOLVER
```

Note that for SIMPLE and SIMPLEC, block 1 is assumed to be velocity. In the above case, the SIMPLEC method is used with the velocity inverse approximated using a DD-ILU preconditioner and the pressure (Schur complement approximation) inverse approximated through algebraic multigrid.

While the SIMPLE and SIMPLEC methods are envisioned for usage in preconditioning Navier–Stokes systems, nothing prevents their usage more generally. The only requirement to testing one of these methods is that the system only contains two sub-block numbers. For systems with more than two sub-blocks, the more general *Block Gauss-Seidel* method is appropriate. If the physics include Navier–Stokes coupled with an additional term, the *Hierarchical Block*

Gauss-Seidel method can be employed.

Hierarchical Block Gauss-Seidel

Note: Hierarchical blocks with a single sub-block must use a preconditioner or solver provided by the `TEKO SUBBLOCK PRECONDITIONER` or `TEKO SUBBLOCK SOLVER` blocks, respectively. For hierarchical blocks with *more than one* sub-block, the `TEKO BLOCK PRECONDITIONER` or `TEKO BLOCK SOLVER` blocks must be used.

As noted in *Navier–Stokes and SIMPLE*, many simulations may include coupling between the Navier–Stokes equations and some other physics. For example, the velocity and pressure components in the Navier–Stokes equations may be coupled with temperature as in the Boussinesq approximation for buoyancy-driven flows. However, it is still possible to use a Teko-based preconditioner that employs the SIMPLE or SIMPLEC methods for the velocity-pressure sub-blocks. This is accomplished by hierarchically decomposing the block-system into smaller blocks, then stitching together the various smaller blocks through a hierarchical block Gauss-Seidel approach. Example syntax for a Teko preconditioner using the hierarchical block Gauss-Seidel is shown below.

```
BEGIN TPETRA EQUATION SOLVER GMRES_TEKO_SIMPLE
  BEGIN GMRES SOLVER
    BEGIN TEKO PRECONDITIONER
      DEFINE MATRIX SUBBLOCK NUMBER=1 DOF=VELOCITY
      DEFINE MATRIX SUBBLOCK NUMBER=2 DOF=PRESSURE
      DEFINE MATRIX SUBBLOCK NUMBER=3 DOF=TEMPERATURE

      BEGIN HIERARCHICAL BLOCK GAUSS-SEIDEL SOLVER
        USE SUBBLOCKS 1,2 FOR HIERARCHICAL BLOCK 1
        USE SUBBLOCKS 3 FOR HIERARCHICAL BLOCK 2
        USE BLOCK INVERSE SIMPLE_SOLVER FOR HIERARCHICAL BLOCK 1
        USE BLOCK INVERSE DDILU_TEMPERATURE_PC FOR HIERARCHICAL BLOCK 2
      END

      BEGIN TEKO BLOCK SOLVER SIMPLE_SOLVER
        BEGIN GMRES SOLVER
          BEGIN TEKO PRECONDITIONER
            DEFINE MATRIX SUBBLOCK NUMBER=1 DOF=VELOCITY
            DEFINE MATRIX SUBBLOCK NUMBER=2 DOF=PRESSURE

            BEGIN TEKO SUBBLOCK PRECONDITIONER DDILU_PC
              BEGIN DD-ILU PRECONDITIONER
              END
```

(continues on next page)

(continued from previous page)

```
            END

            BEGIN TEKO SUBBLOCK PRECONDITIONER AMG_PC
              BEGIN MUELU PRECONDITIONER
              END
            END

            BEGIN SIMPLE SOLVER
              USE INVERSE DDILU_PC FOR SUBBLOCK 1
              USE INVERSE AMG_PC FOR SUBBLOCK 2
              EXPLICIT VELOCITY INVERSE TYPE = ABSROWSUM # or LUMPED,␣
↪DIAGONAL
              ALPHA = 0.9 # Pressure Schur-Complement (under/over)␣
↪relaxation, > 0.0
            END
          END
        END
      END TEKO BLOCK SOLVER

      BEGIN TEKO SUBBLOCK PRECONDITIONER DDILU_TEMPERATURE_PC
        BEGIN DD-ILU PRECONDITIONER
        END
      END

    END
  END
END TPETRA EQUATION SOLVER
```

In the above example, the `HIERARCHICAL BLOCK GAUSS-SEIDEL SOLVER` block is used to decompose the matrix into two main sub-systems: one containing velocity and pressure, the other temperature. The velocity and pressure terms are solved via the SIMPLE method defined in a `TEKO BLOCK SOLVER` block. The temperature, on the other hand, is only a single block. In this case, the `TEKO SUBBLOCK PRECONDITIONER` block is used to specify a DD-ILU preconditioner to approximate the inverse of the temperature block. Note, however, that if we had lumped an additional block with temperature, either the `TEKO BLOCK SOLVER` or `TEKO BLOCK PRECONDITIONER` blocks are required to invert the given block.

Preconditioned Sub-block Solvers

In many scenarios, there may exist a sub-block that is difficult to solve. This is especially problematic if there exists one sub-block that is significantly tougher than the others, as the sub-block that is difficult to solve may lead to stagnation in the overall linear solver convergence. (For information on detecting this scenario, refer to *Diagnostic Sub-block Solvers*). Fortunately, Teko provides options to use any preconditioned iterative solver defined in a `TPETRA EQUATION SOLVER` block for the sub-block solvers to mitigate this issue.

An example input deck expanding on our previous example input for *Block Gauss-Seidel* is provided below.

```
BEGIN TPETRA EQUATION SOLVER GMRES_TEKO_BGS
  BEGIN GMRES SOLVER
    BEGIN TEKO PRECONDITIONER
      DEFINE MATRIX SUBBLOCK NUMBER=1 DOF=VELOCITY
      DEFINE MATRIX SUBBLOCK NUMBER=2 DOF=PRESSURE

      BEGIN TEKO SUBBLOCK SOLVER DDILU_SOLVER
        BEGIN GMRES SOLVER
          CONVERGENCE TOLERANCE = 1e-4
          MAXIMUM ITERATIONS = 30
          RESTART ITERATIONS = 30
          BEGIN DD-ILU PRECONDITIONER
          END
        END
      END

      BEGIN TEKO SUBBLOCK SOLVER AMG_SOLVER
        BEGIN GMRES SOLVER
          CONVERGENCE TOLERANCE = 1e-4
          MAXIMUM ITERATIONS = 30
          RESTART ITERATIONS = 30
          BEGIN MUELU PRECONDITIONER
          END
        END
      END

      BEGIN BLOCK GAUSS-SEIDEL SOLVER
        USE INVERSE DDILU_SOLVER FOR SUBBLOCK 1
        USE INVERSE AMG_SOLVER FOR SUBBLOCK 2
      END
    END
  END
END TPETRA EQUATION SOLVER
```

In the above example, a GMRES iteration seeking to reduce the residual by 4 orders of magnitude with a maximum of 30 iterations is used as the solver for the velocity and pressure sub-blocks. The velocity solve is preconditioned using DD-ILU from Ifpack2, while the pressure is preconditioned using algebraic multigrid from MueLu. The use of an iterative solver for the velocity and pressure sub-blocks means that the Block Gauss-Seidel preconditioner does *not* remain constant with each iteration. This requires the use of a *flexible* solver, such as flexible GMRES [46], to account for this. Aria automatically switches from GMRES to flexible GMRES when using a Teko-based preconditioner.

The use of preconditioned sub-block solves is not limited to Block Gauss-Seidel. An iterative solver can be used to invert the velocity and Schur complement pressure blocks in the SIMPLE and SIMPLEC methods from *Navier–Stokes and SIMPLE*. This is easy to accomplish, as it requires only altering the sub-block solver specification. Extending our example from the *Navier–Stokes and SIMPLE* section, we have:

```
BEGIN TPETRA EQUATION SOLVER GMRES_TEKO_SIMPLE
  BEGIN GMRES SOLVER
    BEGIN TEKO PRECONDITIONER
      # NOTE: velocity must be assigned to block 1 for SIMPLE
      DEFINE MATRIX SUBBLOCK NUMBER=1 DOF=VELOCITY
      DEFINE MATRIX SUBBLOCK NUMBER=2 DOF=PRESSURE

      BEGIN TEKO SUBBLOCK SOLVER DDILU_SOLVER
        BEGIN GMRES SOLVER
          CONVERGENCE TOLERANCE = 1e-4
          MAXIMUM ITERATIONS = 30
          RESTART ITERATIONS = 30
          BEGIN DD-ILU PRECONDITIONER
          END
        END
      END

      BEGIN TEKO SUBBLOCK SOLVER AMG_SOLVER
        BEGIN GMRES SOLVER
          CONVERGENCE TOLERANCE = 1e-4
          MAXIMUM ITERATIONS = 30
          RESTART ITERATIONS = 30
          BEGIN MUELU PRECONDITIONER
          END
        END
      END

      BEGIN SIMPLE SOLVER
        USE INVERSE DDILU_SOLVER FOR SUBBLOCK 1
        USE INVERSE AMG_SOLVER FOR SUBBLOCK 2
```

(continues on next page)

(continued from previous page)

```
        EXPLICIT VELOCITY INVERSE TYPE = ABSROWSUM # or LUMPED, DIAGONAL
        ALPHA = 0.9 # Pressure Schur-Complement (under/over) relaxation, >
↪ 0.0
      END
    END
  END
END TPETRA EQUATION SOLVER
```

Expert Teko Syntax

While the syntax provided through the `TEKO PRECONDITIONER` is meant to cover most use-cases, it is nevertheless helpful to have access to the underlying Teko XML specification through the `EXPERT TEKO PRECONDITIONER` block. For example, the *Diagnostic Sub-block Solvers* currently require the use of the `EXPERT TEKO PRECONDITIONER` block. To demonstrate the `EXPERT TEKO PRECONDITIONER` syntax, we consider converting the Block Gauss-Seidel based Teko solver in *Preconditioned Sub-block Solvers*. An example input deck is provided below.

```
BEGIN TPETRA EQUATION SOLVER GMRES_TEKO_BGS
  BEGIN GMRES SOLVER
    BEGIN EXPERT TEKO PRECONDITIONER
      DEFINE MATRIX SUBBLOCK NUMBER=1 DOF=VELOCITY
      DEFINE MATRIX SUBBLOCK NUMBER=2 DOF=PRESSURE

      INVERSE METHOD = BGS
      XML FILE = teko.xml
    END
  END
END TPETRA EQUATION SOLVER
```

In exchange for removing the block solver specification, the `EXPERT TEKO PRECONDITIONER` block syntax requires specifying the block inverse through the `INVERSE METHOD` command and input XML file through the `XML FILE` command. We recall that the solver specified in the *Preconditioned Sub-block Solvers* example uses a Block Gauss-Seidel solver with a DD-ILU preconditioned GMRES solver for the velocity sub-block and a MueLu preconditioned GMRES solver for the pressure sub-block. The equivalent Teko XML that implements these solver settings is defined below:

```
<ParameterList>
  <ParameterList name="BGS">
    <Parameter name="Type" type="string" value="Block Gauss-Seidel"/>

    <!-- Velocity -->
```

(continues on next page)

(continued from previous page)

```
    <Parameter name="Inverse Type 1" type="string" value="gmres"/>
    <Parameter name="Preconditioner Type 1" type="string" value="ddilu"/>

    <!-- Pressure -->
    <Parameter name="Inverse Type 2" type="string" value="gmres"/>
    <Parameter name="Preconditioner Type 2" type="string" value="amg"/>
  </ParameterList>

  <ParameterList name="ddilu">
    <Parameter name="Type" type="string" value="Ifpack2"/>
    <Parameter name="Prec Type" type="string" value="SCHWARZ"/>
    <ParameterList name="Ifpack2 Settings">
      <Parameter name="schwarz: use reordering" type="bool" value="true"/>
      <Parameter name="subdomain solver name" type="string" value="RILUK"/
↪>
      <ParameterList name="subdomain solver parameters">
       <Parameter name="fact: iluk level-of-fill" type="int" value="1"/>
      </ParameterList>
    </ParameterList>
  </ParameterList>

  <ParameterList name="amg">
    <Parameter        name="Type"                                type="string
↪"   value="MueLu"/>
    <Parameter        name="verbosity"                           type="string
↪"   value="high"/>
    <Parameter        name="coarse: max size"                    type="int" ␣
↪    value="1000"/>
    <Parameter        name="max levels"                          type="int" ␣
↪    value="4"/>
    <Parameter        name="smoother: type"                      type="string
↪"   value="CHEBYSHEV"/>
    <ParameterList    name="smoother: params">
       <Parameter name="chebyshev: degree"                       type="int" ␣
↪    value="4"/>
       <Parameter name="chebyshev: ratio eigenvalue"             type="double
↪"   value="20"/>
       <Parameter name="chebyshev: min eigenvalue"               type="double
↪"   value="1.0"/>
       <Parameter name="chebyshev: zero starting solution"       type="bool"␣
↪    value="true"/>
       <Parameter name="chebyshev: eigenvalue max iterations" type="int" ␣
```

(continues on next page)

(continued from previous page)

```
↪    value="15"/>
    </ParameterList>
    <Parameter        name="aggregation: type"                type="string
↪"   value="uncoupled"/>
    <Parameter        name="aggregation: drop scheme"         type="string
↪"   value="distance laplacian"/>
    <Parameter        name="aggregation: drop tol"            type="double
↪"   value="0.01"/>

    <Parameter        name="repartition: enable"              type="bool"␣
↪    value="true"/>
    <Parameter        name="repartition: min rows per proc"   type="int" ␣
↪    value="1000"/>
    <Parameter        name="repartition: start level"         type="int" ␣
↪    value="2"/>
    <Parameter        name="repartition: max imbalance"       type="double
↪"   value="1.327"/>
    <Parameter        name="repartition: partitioner"         type="string
↪"   value="zoltan2"/>
  </ParameterList>

  <ParameterList name="gmres">
    <Parameter name="Type" type="string" value="Belos"/>
    <Parameter name="Solver Type" type="string" value="Pseudo Block GMRES
↪"/>
    <ParameterList name="Solver Types">
      <ParameterList name="Pseudo Block GMRES">
        <Parameter name="Maximum Iterations" type="int" value="30"/>
        <Parameter name="Num Blocks" type="int" value="30"/>
        <Parameter name="Convergence Tolerance" type="double" value="1e-4
↪"/>
      </ParameterList>
    </ParameterList>
  </ParameterList>
</ParameterList>
```

For using SIMPLE, on the other hand, the input XML may resemble the following:

```
<ParameterList>
  <!-- NOTE: must alter Aria input to specify INVERSE METHOD = SIMPLEC -->
  <ParameterList name="SIMPLEC">
    <Parameter name="Type" type="string" value="NS SIMPLE"/>
```

(continues on next page)

(continued from previous page)

```
    <!- Switch to "Diagonal" for SIMPLE -->
    <Parameter name="Explicit Velocity Inverse Type" type="string" value=
↪"AbsRowSum"/>
    <Parameter name="Alpha" type="double" value="0.9"/>

    <!-- Velocity -->
    <Parameter name="Inverse Velocity Type" type="string" value="gmres"/>
    <Parameter name="Preconditioner Velocity Type" type="string" value=
↪"ddilu"/>

    <!-- Pressure -->
    <Parameter name="Inverse Pressure Type" type="string" value="gmres"/>
    <Parameter name="Preconditioner Pressure Type" type="string" value=
↪"amg"/>
  </ParameterList>

  <ParameterList name="ddilu">
    <Parameter name="Type" type="string" value="Ifpack2"/>
    <Parameter name="Prec Type" type="string" value="SCHWARZ"/>
    <ParameterList name="Ifpack2 Settings">
      <Parameter name="schwarz: use reordering" type="bool" value="true"/>
      <Parameter name="subdomain solver name" type="string" value="RILUK"/
↪>
      <ParameterList name="subdomain solver parameters">
       <Parameter name="fact: iluk level-of-fill" type="int" value="1"/>
      </ParameterList>
    </ParameterList>
  </ParameterList>

  <ParameterList name="amg">
    <Parameter        name="Type"                          type="string
↪"   value="MueLu"/>
    <Parameter        name="verbosity"                     type="string
↪"   value="high"/>
    <Parameter        name="coarse: max size"              type="int" ␣
↪    value="1000"/>
    <Parameter        name="max levels"                    type="int" ␣
↪    value="4"/>
    <Parameter        name="smoother: type"                type="string
↪"   value="CHEBYSHEV"/>
    <ParameterList    name="smoother: params">
       <Parameter name="chebyshev: degree"                 type="int" ␣
```

(continues on next page)

(continued from previous page)

```
↪    value="4"/>
      <Parameter name="chebyshev: ratio eigenvalue"          type="double
↪"   value="20"/>
      <Parameter name="chebyshev: min eigenvalue"            type="double
↪"   value="1.0"/>
      <Parameter name="chebyshev: zero starting solution"    type="bool"␣
↪    value="true"/>
      <Parameter name="chebyshev: eigenvalue max iterations" type="int" ␣
↪    value="15"/>
    </ParameterList>
    <Parameter        name="aggregation: type"               type="string
↪"   value="uncoupled"/>
    <Parameter        name="aggregation: drop scheme"        type="string
↪"   value="distance laplacian"/>
    <Parameter        name="aggregation: drop tol"           type="double
↪"   value="0.01"/>

    <Parameter        name="repartition: enable"             type="bool"␣
↪    value="true"/>
    <Parameter        name="repartition: min rows per proc"  type="int" ␣
↪    value="1000"/>
    <Parameter        name="repartition: start level"        type="int" ␣
↪    value="2"/>
    <Parameter        name="repartition: max imbalance"      type="double
↪"   value="1.327"/>
    <Parameter        name="repartition: partitioner"        type="string
↪"   value="zoltan2"/>
  </ParameterList>

  <ParameterList name="gmres">
    <Parameter name="Type" type="string" value="Belos"/>
    <Parameter name="Solver Type" type="string" value="Pseudo Block GMRES
↪"/>
    <ParameterList name="Solver Types">
      <ParameterList name="Pseudo Block GMRES">
        <Parameter name="Maximum Iterations" type="int" value="30"/>
        <Parameter name="Num Blocks" type="int" value="30"/>
        <Parameter name="Convergence Tolerance" type="double" value="1e-4
↪"/>
      </ParameterList>
    </ParameterList>
  </ParameterList>
</ParameterList>
```

Diagnostic Sub-block Solvers

To get additional diagnostics about the residual of the individual inverse approximations, Teko allows for the injection of a diagnostic inverse type. This feature, however, requires the use of the `EXPERT TEKO PRECONDITIONER` syntax as described in *Expert Teko Syntax*. For example, the SIMPLEC XML from *Navier–Stokes and SIMPLE* is modified to inject diagnostic blocks inside the pressure and velocity sub-block solvers below.

```
<ParameterList name="SIMPLEC">
  <Parameter name="Type" type="string" value="NS SIMPLE"/>
  <Parameter name="Inverse Velocity Type" type="string" value="diagnostic_
↪velocity_block"/>
  <Parameter name="Inverse Pressure Type" type="string" value="diagnostic_
↪pressure_block"/>
  <Parameter name="Explicit Velocity Inverse Type" type="string" value=
↪"AbsRowSum"/>
</ParameterList>

<!--
  See sample XML in 'EXPERT TEKO SYNTAX' section for definitions of the␣
↪ddilu, amg sub-block solvers
-->
<ParameterList name="diagnostic_pressure_block">
  <Parameter name="Type" type="string" value="Diagnostic Inverse"/>
  <Parameter name="Inverse Factory" type="string" value="amg"/>
  <Parameter name="Descriptive Label" type="string" value="Pressure"/>
  <Parameter name="Print Residual" type="bool" value="true"/>
</ParameterList>

<ParameterList name="diagnostic_velocity_block">
  <Parameter name="Type" type="string" value="Diagnostic Inverse"/>
  <Parameter name="Inverse Factory" type="string" value="ddilu"/>
  <Parameter name="Descriptive Label" type="string" value="Velocity"/>
  <Parameter name="Print Residual" type="bool" value="true"/>
</ParameterList>
```

The above XML file will output the relative residual of the sub-block solvers, for example:

```
DiagnosticLinearOp "Velocity": residual = [ 2.0547e-03 ]
DiagnosticLinearOp "Pressure": residual = [ 3.8366e-01 ]
```

Injecting diagnostic inverses, like in the example above, is helpful for determining which portion of a block preconditioning scheme is failing to converge. For blocks that are not able to converge or do not converge as quickly, consider using an iterative solver for the block as outlined in *Preconditioned Sub-block Solvers*. *Road-map for Constructing Teko Solvers* offers further advice

on debugging block preconditioner convergence.

Road-map for Constructing Teko Solvers

As we have seen from the previous sections, there is a lot of upfront work in setting up a Teko preconditioner. Constructing a Teko preconditioner for a multiphysics simulations requires the combination of a good block-splitting, block inverse method, and sub-block solvers. This section aims to offer advice to the user in how to construct a Teko preconditioner.

The following steps provide a road-map to constructing a Teko solver.

1. As a starting point, gather the names of the DOFs in the linear system you wish to solve.
   - Check what EQs are defined in the equation system where you want to use a Teko solver.
   - If you are still unsure what DOFs are in the linear system, guess! The error message in the logfile will provide the names for each DOF in the linear system.
2. Set up the block preconditioner.
   - Start with mapping each DOF to a unique sub-block.
   - If possible, use a direct solver for each sub-block to start.
   - For Navier–Stokes, start with the SIMPLE or SIMPLEC method from *Navier–Stokes and SIMPLE*.
   - For general multiphysics problems with only 2 sub-blocks, consider using SIMPLE or SIMPLEC. Note that the order of the sub-blocks matter as sub-block 2 is used to form the approximate Schur complement solve.
   - For general multiphysics problems with 3 or more sub-blocks, use *Block Gauss-Seidel*.
3. Find a good block-splitting by altering the DOF-to-block mapping.
   - Start with mapping each DOF to a unique sub-block.
   - If possible, use a direct solver for the sub-blocks to remove the concern for finding good sub-block solvers. We will tackle that issue on step 4. *Note* that the block-splitting with direct solvers forms a lower bound on the iteration count. If the iterations are relatively high – even 10 to 20 iterations is a lot here – it is unlikely the solver will perform well once we get to step 4.
   - Use physical intuition to lump DOFs to the same block. DOFs with tight-coupling are best put in the same sub-block. A reasonable starting point is to lump ‘like-with-like’. For example, try assigning species equations in the same phase to the same sub-block. Lumping temperature equations across phases tends to do well.
   - This step will almost always require some amount of trial-and-error.
4. Find good sub-block solvers.

- We used direct solvers for steps 2 and 3 as a stepping stone. Now, we should try to remove their usage.
- Sub-blocks encompassing several bits of physics often require DD-ILU.
- Sub-blocks that only comprise a Poisson or elasticity equation are good candidates for algebraic multigrid and MueLu.
- Sub-blocks with solid phase densities or solid mass fractions may be solved using Jacobi.
- If in doubt, DD-ILU is a reasonable first choice for a sub-block solver.
- Use the diagnostic sub-block solvers from *Diagnostic Sub-block Solvers* to monitor convergence. You generally want each sub-block to reach at least 1-2 digits of residual reduction. Be on the lookout for any sub-blocks that exhibit significantly worse convergence. These may lead to stagnation in the linear solver.
- Blocks that exhibit significantly worse convergence may benefit from using a Belos solver as in *Preconditioned Sub-block Solvers*.
- If all else fails, leave the direct solver in place for a sub-block.

Following the advice from steps 1-4, you should now be able to construct a Teko preconditioner. For an example demonstrating setting up a Teko solver, see *Teko Solver for OMD with Porous Flows Example*.

4.8 Controlling the Simulation

The present section describes how to control the progression of a simulation. Specifically, *Solution Control* discusses how to set up a solution control description to progress the solution, *Solution Options* discusses options that augment the behavior of the simulation, and *Time Step Selection* provides details and guidance on how to select a time-stepping strategy and value.

Solution Control

All Aria input files must include a Solution Control Description block in the Procedure section of the input file. This description contains directives for executing a steady-state (sequential) or transient analysis ranging from simple single-region analyses to more complicated multi-region execution including transfer, nested nonlinear iterations, and/or subcycling.

A solution control block has three parts:

System
: Controls how the simulation is advanced

Parameters
: Parameters to inform how each region is handled by `system`

Initialization
: Additional actions to do at startup before `system`-defined behavior

See the *command reference* for details on all possible commands.

The system block controls how the analysis progresses by `advance`ing the relevant region(s). For a *Transient Analysis* this corresponds to taking a step, whereas for a *Steady Analysis* this corresponds to solving the steady problem (i.e. taking the ONLY step).

When there are multiple regions involved in an analysis, it is often necessary to transfer information between them. Once a *transfer block* is set up, this is accomplished with the `transfer` directive. These more complex analyses may also require explicitly *initialization* of fields. Depending on the simulation types, various *parameters* are required, as is demonstrated in the sections below.

Typically, regions control their own postprocessing when they during `advance`. In rare cases, one may wish to trigger the postprocessors manually (e.g. *adaptivity*). To do so, one can explicitly `postprocess` at any point in a solution control block.

Steady Analysis

To set up a simple steady analysis, a *Sequential* block can be used to `advance` a region. As a simple example

Simple Sequential Input

```
Begin Sierra myJob

  # Materials, Solvers, Finite Element Model, etc
  ...

  Begin Procedure myProcedure

    Begin Solution Control Description
      # Note that the description calls out the system to use
      Use System Main
      Begin System Main
        Begin Sequential MySolveBlock
          Advance steadyRegion
        End
      End
    End

    Begin Aria Region steadyRegion
      # ICs, BCs, equations, output, etc
      ...
```

(continues on next page)

(continued from previous page)

```
    End Aria Region steadyRegion

    ...
  End Procedure myProcedure
End Sierra myJob
```

Though its often more favorable to solve fully-coupled physics within a single application, this is not always possible. As an alternative, loose-coupling is often carried out by exchanging solution information across multiple applications. Solution control provides various means of carrying out these analyses depending upon the strength of physics coupling within a solution step.

Note: This section refers to coupling *within* the Sierra ecosystem, e.g. coupling Aria and Adagio via Arpeggio. For weak coupling with apps outside of the Sierra ecosystem, see *File Based Coupling*. For stronger coupling, see *MPMD Coupling*.

When the different problem physics are weakly-coupled, one is assuming that for each solution step it is sufficient to supply current values of variables involved in the coupling. In this case, the regions can be sequentially `advance`d with information exchanged in between via `transfer`s. Below, an example of a weakly coupled Aria-Adagio simulation is provided

Weakly Coupled Sequential Input

```
Begin Sierra myJob

  # Materials, Solvers, Finite Element Model, etc
  ...

  Begin Procedure myProcedure

    Begin Solution Control Description
      Use System Main
      Begin System Main
        Begin Sequential MySolveBlock
          Advance ariaRegion
          transfer aria_to_adagio
          Advance adagioRegion
          transfer adagio_to_aria
        End
      End
    End

    Begin Aria Region ariaRegion
      # ICs, BCs, equations, output, etc
      ...
```

(continues on next page)

(continued from previous page)

```
    End Aria Region ariaRegion

    Begin Adagio Region adagioRegion
      # ICs, BCs, equations, output, etc
      ...
    End Adagio Region adagioRegion

    Begin transfer adagio_to_aria
      # transfer commands
      ...
    End transfer adagio_to_aria

    Begin transfer aria_to_adagio
      # transfer commands
      ...
    End transfer aria_to_adagio

    ...
  End Procedure myProcedure
End Sierra myJob
```

For a stronger coupling, one may chose to iterate within a solution step until the interaction between physics has converged. This is accomplished by specifying a *Nonlinear* command block within the sequential block. Here the nonlinear iteration will continue until user specified criteria *Parameters for Nonlinear Iteration* is satisfied

Stronger Coupled Sequential Input

```
Begin Sierra myJob

  # Materials, Solvers, Finite Element Model, etc
  ...

  Begin Procedure myProcedure

    Begin Solution Control Description
      Use System Main
      Begin System Main
        Begin Sequential MySolveBlock
          Begin Nonlinear Iteration
            Advance ariaRegion
            transfer aria_to_adagio
            Advance adagioRegion
```

(continues on next page)

(continued from previous page)

```
            transfer adagio_to_aria
          End
        End
      End
    End

    Begin Parameters for Nonlinear Iteration
      Converged when "adagioRegion.MaxInitialNonlinearResidual(0) < 1E-1 &
↪& \$
                      ariaRegion.MaxInitialNonlinearResidual(0) < 1E-1 "
    End

    Begin Aria Region ariaRegion
      # ICs, BCs, equations, output, etc
      ...
    End Aria Region ariaRegion

    Begin Adagio Region adagioRegion
      # ICs, BCs, equations, output, etc
      ...
    End Aria Region adagioRegion

    Begin transfer adagio_to_aria
      # transfer commands
      ...
    End transfer adagio_to_aria

    Begin transfer aria_to_adagio
      # transfer commands
      ...
    End transfer aria_to_adagio

    ...
  End Procedure myProcedure
End Sierra myJob
```

Transient Analysis

Similar to a sequential block, a *Transient* block defines the way transient region(s) advance. A simple example of a transient solution control block will look similar to that of a sequential analog, except that additional *parameters* are required to describe the time range of the simulation. Additionally, parameters defining the time integration for each Region must also be supplied by the user. The example below demonstrates both a `fixed` and `adapative` timestep transient. See *Time Step Selection* for guidance on setting these parameters.

Simple Transient Input

```
Begin Sierra myJob

  # Materials, Solvers, Finite Element Model, etc
  ...

  Begin Procedure My_Aria_Procedure

    Begin Solution Control Description

      Use System Main

      Begin System Main
        Simulation Start Time            = 0.0
        Simulation Termination Time      = 10.0
        Simulation Max Global Iterations = 1000

        Begin Transient Time_Block_1
          Advance My_Aria_Region
        End
        Begin Transient Time_Block_2
          Advance My_Aria_Region
        End
      End

      #Parameters for Time_Block_1 behavior
      Begin Parameters For Transient Time_Block_1
        Start Time      = 0.0
        Number of steps = 8
        Begin Parameters For Aria Region My_Aria_Region
          Time Step Variation    = Fixed
          Initial Time Step Size = 0.001
        End
      End
```

(continues on next page)

(continued from previous page)

```
    #Parameters for Time_Block_2 behavior
    Begin Parameters For Transient Time_Block_2
      Begin Parameters For Aria Region My_Aria_Region
        Time Step Variation           = Adaptive
        Initial Time Step Size        = 0.001
        Predictor-Corrector Tolerance = 1e-3
        Minimum Time Step Size        = 1e-6
      End
    End

    ...
  End Procedure My_Aria_Procedure
End Sierra myJob
```

Note: Note that for each named `Begin Transient [NAME]` command block there should be a corresponding `Begin Parameters for Transient [NAME]` command block which defines its parameters.

Similar to the sequential case, multiple regions can be tied together either in a *weak* or *strong* coupling using `transfer`s and *Nonlinear* command blocks. The stronger case is demonstrated below. Here, the time step for advancement to the next solution step is negotiated between the coupled Regions based upon their respective Transient parameters.

Coupled Transient Input

```
Begin Sierra myJob

  # Materials, Solvers, Finite Element Model, etc
  ...

  Begin Procedure My_Aria_Procedure

    Begin Solution Control Description

      Use System Main

      Begin System Main
        Simulation Start Time            = 0.0
        Simulation Termination Time      = 10.0
        Simulation Max Global Iterations = 1000
```

(continues on next page)

(continued from previous page)

```
    Begin Transient Time_Block
        Begin Nonlinear Iteration
          Advance myAriaRegion
          transfer aria_to_adagio
          Advance myAdagioRegion
          transfer adagio_to_aria
        End Nonlinear Iteration
    End
  End

  Begin Parameters for Nonlinear Iteration
    Converged when "myAdagioRegion.MaxInitialNonlinearResidual(0) <␣
↪1E-1 && \$
                    myAriaRegion.MaxInitialNonlinearResidual(0) <␣
↪1E-1 "
  End

  Begin Parameters For Transient Time_Block_1
    Start Time      = 0.0
    Termination Time = 10.0

    #Aria region-specific parameters for Time_Block_1
    Begin Parameters For Aria Region myAriaRegion
        Time Step Variation    = adaptive
        Initial Time Step Size = 0.001
        Predictor-Corrector Tolerance = 1e-3
        Minimum Time Step Size        = 1e-6
    End

    #Adagio region-specific parameters for Time_Block_1
    Begin Parameters For Adagio Region myAdagioRegion
        Time increment = 0.001
    End
  End

End Solution Control Description

Begin transfer adagio_to_aria
  # transfer commands
  ...
End transfer adagio_to_aria
```

(continues on next page)

(continued from previous page)

```
    Begin transfer aria_to_adagio
      # transfer commands
      ...
    End transfer aria_to_adagio

    ...
  End Procedure My_Aria_Procedure
End Sierra myJob
```

Note: Note here that a subblock for parameters of each region participating in the transient is present in the `Parameters` block of the transient.

Depending on the two physics being coupled, the two regions can have differing time scales. In this case a *Subcycle* command block can be used to subcycle the region with a smaller timescale, rather than forcing the larger timestep region to follow this smaller step. The solution will advance when the two time applications arrive at the same solution time.

Subcycled Transient Input

```
Begin Sierra myJob

  # Materials, Solvers, Finite Element Model, etc
  ...

  Begin Procedure My_Aria_Procedure

    Begin Solution Control Description

      Use System Main

      Begin System Main
        Simulation Start Time            = 0.0
        Simulation Termination Time      = 10.0
        Simulation Max Global Iterations = 1000

        Begin Transient Time_Block_1
          Transfer Adagio_to_Aria
          Advance My_Aria_Region
          Transfer Aria_to_Adagio
          Begin Subcycle AdagioSubcycle
```

(continues on next page)

(continued from previous page)

```
            Advance AdagioRegion
          End
        End

      End

      Begin Parameters For Transient Time_Block_1
        Start Time      = 0.0
        Number of steps = 8

        Begin Parameters For Aria Region My_Aria_Region
          Time Step Variation    = Fixed
          Initial Time Step Size = 0.001
        End

        Begin Parameters for Adagio Region AdagioRegion
          initial time step = 1.0e-6
          # time step scale factor = 1.0
          time step increase factor = 10.
          # step interval = 500
        End
      End

    End

    Begin transfer adagio_to_aria
      ...
    End transfer adagio_to_aria

    Begin transfer aria_to_adagio
      ...
    End transfer aria_to_adagio

    Begin Aria Region myAriaRegion
      ...
    End Aria Region myAriaRegion

    Begin Adagio Region myAdagioRegion
      ...
    End Adagio Region myAdagioRegion

  End Procedure myProcedure
End Sierra myJob
```

Variable Initialization

In the case of transient analysis it is sometimes necessary to initialize a distribution of values before the analysis actually begins. This can be useful, for example, to transfer a field containing initial conditions to a coupled region. This is accomplished by using an *Initialize* command block. An example of this case is provided below

Initialization Transient Input

```
Begin Sierra myJob

  # Materials, Solvers, Finite Element Model, etc
  ...

  Begin Procedure My_Aria_Procedure

    Begin Solution Control Description

      Use System Main

      begin initialize init
        Advance AdagioRegion
        Transfer Adagio_to_Aria
      end initialize init

      Begin System Main

        # Note that the system calls out the initialize to use
        Use initialize init

        Simulation Start Time            = 0.0
        Simulation Termination Time      = 10.0
        Simulation Max Global Iterations = 1000

        Begin Transient Time_Block_1
          Advance My_Aria_Region
          Transfer Aria_to_Adagio
          Begin Subcycle AdagioSubcycle
            Advance AdagioRegion
          End
          Transfer Adagio_to_Aria
        End

      End
```

(continues on next page)

(continued from previous page)

```
    Begin Parameters For Transient Time_Block_1
      Start Time      = 0.0
      Number of steps = 8

      Begin Parameters For Aria Region My_Aria_Region
        Time Step Variation    = Fixed
        Initial Time Step Size = 0.001
      End

      Begin Parameters for Adagio Region AdagioRegion
        initial time step = 1.0e-6
        # time step scale factor = 1.0
        time step increase factor = 10.
        # step interval = 500
      End
    End

    Begin transfer adagio_to_aria
      ...
    End transfer adagio_to_aria

    Begin transfer aria_to_adagio
      ...
    End transfer aria_to_adagio

    Begin Aria Region myAriaRegion
      ...
    End Aria Region myAriaRegion

    Begin Adagio Region myAdagioRegion
      ...
    End Adagio Region myAdagioRegion

  End Procedure myProcedure
End Sierra myJob
```

Solution Options

Solution options are typically used to modify global settings of the simulation. These options are typically tied to specific use cases, such as *Turbulence Model Specification*

```
# Scope: Sierra > Procedure > Aria Region
Begin Solution Options
  ...
  begin turbulence model specification
    turbulence model = smag
  end
end
```

or *Porous Flow Options*

```
# Scope: Sierra > Procedure > Aria Region
Begin Solution Options
  ...
  Begin Porous Flow Options
    Use CVFEM = true
  End
End
```

Otherwise, they control global behavior of algorithms such as clipping utilities

```
# Scope: Sierra > Procedure > Aria Region
BEGIN SOLUTION OPTIONS
  omit enthalpy adjustment after temperature clipping
  ...
END SOLUTION OPTIONS
```

See the *command reference* for a detailed list of all possible commands.

Time Step Selection

This section discusses the available time stepping approaches and values for a transient simulation. For more information on how to setup this transient simulation, see *Transient Analysis*. See the *command reference* for a complete list of Aria parameters that can be specified.

Initial Time Step Selection

The analysis of a transient diffusion problem requires the selection of a suitable time step for the integration procedure. Aria provides both *Adaptive Time Step Selection* and *Fixed Time Step Selection* functionality. In either case, though, some initial estimate of an appropriate time step is required.

If the initial step is *too large*, a loss of temporal accuracy can occur and produce a nonphysical oscillatory response. Likewise, an inappropriately *small* initial time step may produce nonphysical, spatial oscillations in the early time temperature field due to the limited resolution ability (of temperature gradients) of the finite element mesh. Either of these difficulties may lead to stability problems if the boundary value problem is nonlinear. In any event, the solution during the oscillatory period is not accurate and these occurrences should be avoided.

A reasonable initial time step will typically be a function of the length scale(s) of the problem as well as its material properties. See *Guidelines* for advice on selecting an initial timestep value.

For both fixed and adaptive time stepping, an integration method can optionally be specified by setting the *Time Integration Method*. See the *theory section* for more details on the available integration methods.

Fixed Time Step Selection

The simplest available time stepping strategy is fixed time stepping. As the name implies, this progresses the region at a constant time step until the completion criteria is met.

```
# Scope: Sierra > Procedure > Solution Control Description
Begin Parameters For ...
  ...
  begin parameters for Aria region
    initial time step size = 0.15
    time step variation = fixed

    # optionally specify integration method
    time integration method = bdf2
  end
end
```

Note that Aria may still attempt to adjust after a failed time step A way to ensure that failing the fixed timestep is fatal is by setting this value as the minimum timestep as well

```
# Scope: Sierra > Procedure > Solution Control Description
Begin Parameters For ...
  ...
  begin parameters for Aria region
    # time_step = {time_step = 0.15}
    initial time step size = {time_step}
    minimum time step size = {time_step}
    time step variation = fixed
  end
end
```

Adaptive Time Step Selection

Unlike fixed time stepping, adaptive time stepping allows for the step size to be adjusted dynamically based on user-specified criteria. These criteria are roughly split into two parts, error-based criteria and metric-based criteria. First, the error-based (*Predictor Corrector*) criteria ensures that the converged Newton step solution satisfies a specified error-bound. This is accomplished by estimating the solution error as a comparison to the predicted solution obtained from the previous steps. Aside from the error-based criteria, additional *Metric-Based Criteria* ensure that various aspects of the solution progress as desired. Once all these metrics are defined, the procedure for *Step Size Selection* and *Time Step Failure* is discussed.

Predictor Corrector

Let the predictor corrector tolerance, ϵ, be specified from the input file.

$$\frac{\Delta t_{n+1}}{\Delta t_n} = \left(b \, \frac{\epsilon}{d_{n+1}} \right)^m$$

For 1st order time integrator $m = 1/2$, $b = 2.0$. For 2nd order time integrator $m = 1/3$

$$b = 3 \left(1 + \frac{\Delta t_{n-1}}{\Delta t_n} \right)$$

Then a nodal averaged global predictor error can be computed as

$$d_{n+1} = \frac{1}{U_{max}} \left[\frac{\sum_{i=1}^{N} \left(U_{i(n+1)} - U^p_{i(n+1)} \right)^2}{N} \right]^{1/2}$$

where N is the number of nodes on the mesh and U^p is the predicted solution (See *Predictor-Corrector* for more information on the prediction process).

There is an option to compute a volume weighted global predictor error defined as:

$$d_{n+1} = \frac{1}{U_{max}} \left[\frac{\sum_{i=1}^{N} \left(U_{i(n+1)} - U^{p}_{i(n+1)} \right)^2 V_i}{\sum_{i=1}^{N} V_i} \right]^{1/2}$$

so that the timestep based on predictor criteria is

$$\Delta t_{n+1} \leq \Delta t_{\text{pred}} = \Delta t_n \left(b \, \frac{\epsilon}{d_{n+1}} \right)^m \quad .$$

Where V_i is an averaged volume of the connected elements on node i.

The nodal averaged predictor error is the default method, to turn on the volume weighted predictor error, set the following option in the solution control description:

```
# Scope: Sierra > Procedure > Solution Control Description
Begin Parameters For ...
  ...
  begin parameters for Aria region
    ...
    predictor-corrector weighting = volume_weighted
  end
end
```

A volume weighted approach is the default behavior for problems with mesh adaptivity.

You can also manually turn on nodal averaging using the following command

```
# Scope: Sierra > Procedure > Solution Control Description
Begin Parameters For ...
  ...
  begin parameters for Aria region
    ...
    predictor-corrector weighting = nodal_average
  end
end
```

At least one field must be included in the predictor-based time step calculations. The default is for all fields to be included, in the calculation, but alternatively fields can be explicitly included

```
# Scope: Sierra > Procedure > Aria Region
Begin equation system myEqSys
  predictor fields = solid_phase_voltage
  predictor fields = liquid_phase_voltage

  ...
end
```

or excluded

```
# Scope: Sierra > Procedure > Aria Region
Begin equation system myEqSys
  predictor fields = not solid_phase_voltage
  predictor fields = not liquid_phase_voltage

  ...
end
```

for a given equation system.

Note: The time integration method often dictates how many solution steps into a simulation adaptive time stepping can begin. In most cases one must be able to obtain at least two successive solves using a fixed initial time step before adaptive time stepping begins. In many fluid simulations one way wish to increases this number of constant steps to e.g. three or four using the *Corrector Begin After Step* command line.

Metric-Based Criteria

Max CFL

In some instances the stability of a simulation (e.g. flow, mesh displacement, etc) can be governed by a Courant-Friedrichs-Levy (CFL) number. If the maximum value of the CFL over the domain for some quantity is $\mathrm{CFL}_{\mathrm{max,i}}$ and the user specifies a limit of $\mathrm{CFL}_{\mathrm{lim,i}}$, then the bounding time step can be computed as

$$\Delta t_{n+1} \leq \Delta t_{\mathrm{max,CFL}} = \min_i \left[\frac{\mathrm{CFL}_{\mathrm{lim,i}}}{\mathrm{CFL}_{\mathrm{max,i}}} \Delta t_n \right]$$

See the `COURANT LIMIT` commands in the *command reference* for possible Courant limits to impose. For example

```
# Scope: Sierra > Procedure > Solution Control Description
Begin Parameters For ...
  ...
  begin parameters for Aria region
    Interface Courant Limit = 0.5
    Mesh Courant Limit      = 0.5
    Courant Limit           = 0.5

    ...
```

(continues on next page)

(continued from previous page)

```
  end
end
```

Time Step Size Ratio

When using adaptive time stepping, a time step size ratio r is defined to compare the change in time step

$$r = \frac{\Delta t_{n+1}}{\Delta t_n}$$

where Δt_{n+1} is the proposed time step provided from the adaptive time step selection process and Δt_n is the timestep of the current step.

The time step size ratio is bounded in order to assume the step size does not change abruptly. The maximum value $r_{\max}$ ensures the step does not grow too quickly

$$\Delta t_{n+1} \leq \Delta t_{\text{r,max}} = r_{\max} \Delta t_n \ .$$

Its value will fallback to a default if not given, or can be set using the *Maximum Time Step Size Ratio* command. For example,

```
# Scope: Sierra > Procedure > Solution Control Description
Begin Parameters For ...
  ...
  begin parameters for Aria region
    maximum time step size ratio = 4.2

    ...
  end
end
```

Maximum Solution Increment

This criteria attempts to limit the max nodal increment of some solution DOF b to be less than $(\Delta b)_{\max}$ so that given the rate $\dot{b}$,

$$\Delta t_{n+1} \leq \Delta t_{\text{incr}} = 0.95 * (\Delta b)_{\max} / \dot{b}_{\max}$$

This limit is applied for all DOFs for which a *Limit Solution Increment* command is specified, e.g. for temperature:

```
# Scope: Sierra > Procedure > Solution Control Description
Begin Parameters For ...
  ...
  begin parameters for Aria region
    #Limit change in temperature to 2 K per step
    Limit Solution Increment Temperature = 2

    ...
  end
end
```

Note: Although adaptive time stepping with Solution Increment requires at least one successful solve here we require at least two successful solves since control of solution error is the primary objective.

Known Time Discontinuities

In certain problems, there might be points in time where are abrupt changes to the problem. Examples of this include an abrupt change in boundary condition or applied flux. If these times are known a priori, they can declared so that Aria ensures the discontinuity is hit exactly. If the next known discontinuity will occur at $t_{disc,next}$, then

$$\Delta t_{n+1} \leq \Delta t_{\text{disc}} = t_{disc,next} - t_n$$

These times are declared with the *Known Time Discontinuities* line command as is demonstrated in the example below

```
# Scope: Sierra > Procedure > Solution Control Description
Begin Parameters For ...
  ...
  begin parameters for Aria region
    # List of known discontinous times
    Known Time Discontinuities = 1.2 3.4 5.6

    ...
  end
end
```

Step Size Selection

With all conditions defined, the new time step is then determined as the largest step which satisfies all criteria, i.e. for all $\Delta t_{n+1} \leq \Delta t_{\text{lim,i}}$, the new step is determined by

$$\Delta t_{n+1} = \min_i \Delta t_{\text{lim,i}}$$

This calculated step is then optionally saturated by a *minimum* and/or *maximum* time step size $\Delta t_{\text{min}} \leq \Delta t_{n+1} \leq \Delta t_{\text{max}}$ e.g.

```
# Scope: Sierra > Procedure > Solution Control Description
Begin Parameters For ...
  ...
  begin parameters for Aria region
    Minimum Time Step Size = 1e-6
    Maximum Time Step Size = 1e-3

    ...
  end
end
```

If *additional output times* are requested with a `Timestep adjustment interval`, this step might be modified slightly to exactly hit the requested time.

Note: Certain model features (e.g. Chemistry) will require that the behavior within the current time step be considered when advancing the time step. When adaptive time step selection is used, the subcycled chemistry timestep cannot be included directly in the adaptive time step selection as the chemistry time step will seldom satisfy the time step criteria. Here the minimum subcycled chemistry time step is modified to be comparable with other adaptive time step criteria. See *Chemistry* for more information.

The overall time stepping procedure is demonstrated in Fig. 4.29.

Time Step Failure

Aria transient simulations are monitored for time step failure. For each solution time step, we expect that the specified nonlinear tolerance will be achieved in at most the permitted number of nonlinear iterations of the Newton step. If the convergence condition is not met, the time step is said to have "Failed" and solution time will not be allowed to advance.

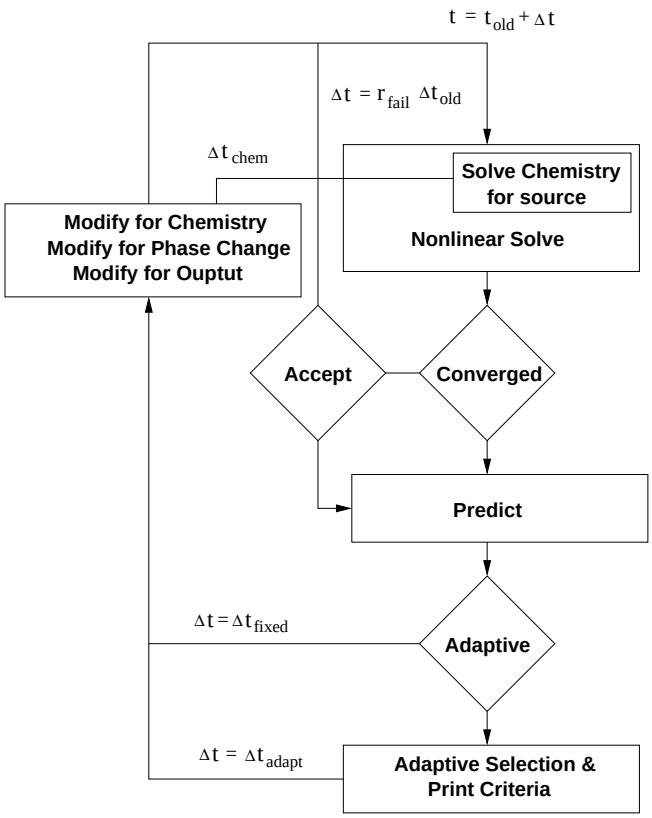


Fig. 4.29: Time Step Selection Schematic

Note: The exception to this is when the user has explicitly stated that failed time steps should be accepted with the *Accept Solution After Maximum Nonlinear Iterations* command line.

If the non-converged solution is not accepted, a failed time step counter is incremented, and the time step is reduced by a factor r_{fail} set by the *Failed Time Step Size Ratio* command

$$\Delta t_{n+1} = r_{\text{fail}} \Delta t_n \ .$$

Rather than allow a simulation to continually fail until hitting the minimum step, the *Maximum Consecutive Time Step Failures* command can be set to end the simulation after a specified number of consecutive failures. See *Repeated Time Step Failure* for troubleshooting these timestep failures.

During phase change, the model using the energy equation for temperature employs a mushy zone formulation hence it is important that the temperature field values not bypass the range of the mushy zone so that the heat of fusion be accounted for. In this case an additional algorithm is used to detect when a nodal temperature has bypassed the mushy zone. When the mushy zone has been bypassed within a timestep, the Newton step will be marked as failed and the current time step will be scaled by a user specified `PHASE CHANGE RELAXATION FACTOR` in the same way that the failed time step size ratio is applied

```
# Scope: Sierra > Procedure > Aria Region
Begin equation system myEqSys
  PHASE CHANGE RELAXATION FACTOR = 0.75

  ...
end
```

In some simulations a user may wish to interrupt the Newton step (fail the step) based upon the residual behavior. For many problems an aberrant behavior is recognized early on by noting the ratio of the current residual to the first residual. In cases where this ratio continues to grow it may be appropriate to terminate the Newton step and re-try with a different time step. This ratio can be specified by the user with the *Maximum Linear Residual Ratio* command.

Note: Failure of a time step due to violation of time step size ratio could be due to the adaptive time stepping selection criteria. If this is frequently encountered, one should consider whether any of the selection criteria values need to be modified in the time step parameters command block(s).

Note: In problems where the adaptive time step drops dramatically at some point in the simulation, the criteria previously described may cause the time step to fail repeatedly before the simulation step finally advances. This can notably increase the overall simulation time. If these drops happen at predictable times, they can be added as *Known Time Discontinuities*. Otherwise, the time step size ratio criteria can be reduced to permit more abrupt changes.

4.9 Input/Output

Output Reference

Output is divided into two major categories:

- *Results Output* contains binary format simulation results in a form suitable for visualization. While the Results output usually corresponds to the entire model, it can also be applied to portions of the model.
- *Heartbeat Output* is generally written to a text file and provides a convenient means of monitoring intermediate simulation results as well as outputting *reduction variables*.

This section describes how to use these two output functionalities. A detailed list of all possible commands can be found in the *command reference*.

Results Output

The results output block allows you to print simulation data to an output database. This database can then be visualized/postprocessed (e.g. in ParaView), used in subsequent simulations, etc. This data includes *internal fields* such as `SOLUTION->DOF`, *user fields*, or *postprocessed quantities* as well as *global variables*.

A list of the quantities available for output is printed to the log immediately after the mesh is read. This list can be previewed by running aria with the `--check-input` option which will verify the input syntax, read the mesh, print the variables available for output, and then exit.

```
Aria Region "myRegion" has the following fields available for output:
  NODE_RANK Fields:
    * anti_time_derivative_at_time->TEMPERATURE
    * cht_H
    * cht_Too
    * model_coordinates
    * physical_coordinates
    * pp->HEAT_FLUX
    * predicted_solution->TEMPERATURE
```

(continues on next page)

(continued from previous page)

```
    * solution->TEMPERATURE
    * solution->TEMPERATURE_STKFS_N
    * solution->TEMPERATURE_STKFS_NM1
    * time_derivative_at_time->TEMPERATURE
    * ...
  EDGE_RANK Fields:
    * face_coverage
  ELEMENT_RANK Fields:
    * current_element_volume
    * initial_element_volume
    * volume_change_ratio
  Global Variables:
    * avg_temperature
    * dt
    * max_temperature
    * number_of_elements
    * number_of_nodes
    * time
```

The available quantities are split by their entity rank. When specifying a field for output, the associated rank should be used. A quantity is added to the output database using a line of the form

```
<entity> variables = <internal name> [as <output name>]
```

where the `entity` is the rank specified in the output list, and the `internal name` can optionally be overridden in the output database by specifying an `output name` e.g.

```
# Scope: Sierra > Procedure > Aria Region
begin results output myOutput
  # name of the output database
  Database name = results.e

  # Frequency information
  at step 0 increment = 1

  #Quantities to print
  nodal variables = someMissingField
  nodal variables = solution->TEMPERATURE as temp
  element variables = volume_change_ratio
  global variables = avg_temperature
end
```

Note: It is worth noting that since the ExodusII format has no concept of vector/tensor quantities, Paraview deduces vectors by looking for the same variable with `X`, `Y`, and `Z` (for 3D problems) appended. For example if `Ux`, `Uy`, and `Uz` are all present in an output database, ParaView will group them into a vector called `U`. Similarly if a field called `displacement` is found, it will be recognized by ParaView as the nodal displacements and the mesh will automatically be warped by the field. Aria will handle naming of vector/tensor quantities that are requested for output, but it is worth keeping these facts in mind when specifying your own output names.

Note: It is not fatal to request an output quantity that does not exist. Instead, a warning is printed to the log stating that output is skipped since the field cannot be found. If a certain quantity is missing from your output, this can be a good first thing to check for.

```
*****************************************************************************
**  Warning:
**  In region 'myRegion' Results Output block 'myOutput',␣
↪Nodal variable
**  'someMissingField' was not found and will not be output
*****************************************************************************
```

Heartbeat Output

Unlike *Results Output*, heartbeats are used to view scalar data such as reduced quantities and running metrics. They can be useful to track progress of simulations, as well as for collecting reduced data for postprocessing.

Available output quantities include

- `TIME/TIMESTEP` - Current simulation time/timestep
- `STEP` - Current simulation step
- `NUMBER_OF_ELEMENTS/NUMBER_OF_NODES` - Mesh statistics
- `KILLED_ELEMENTS/TOTAL_DEAD_ELEMENTS` - *Standard Element Death* info
- Any postprocessed *global variable*

As was stated above, the list of the available globals will be printed to the log

```
Aria Region "myRegion" has the following fields available for output:
  NODE_RANK Fields:
    * ...
```

(continues on next page)

(continued from previous page)

```
  EDGE_RANK Fields:
    * ...
  ELEMENT_RANK Fields:
    * ...
  Global Variables:
    * avg_temperature
    * dt
    * max_temperature
    * number_of_elements
    * number_of_nodes
    * time
```

A quantity is added to the output heartbeat using a line of the form

```
variable = <entity> <internal name> [as <output name>]
```

Note that similar to results output, the `internal name` can be overridden by specifying `as <output name>`

```
# Scope: Sierra > Procedure > Aria Region
Begin Heartbeat myHb
  # name of the output stream
  Stream name = results.csv

  #Formatting of file
  Timestamp Format = ""
  Precision        = 8   # precision of output
  Format           = csv # delimeter
  Labels           = Off # qnty=123 format
  Legend           = On  # header row of labels

  # Frequency information
  at step 0 increment = 1

  #Quantities to print
  variable = global time as t
  variable = global avg_temperature
  variable = global max_temperature
  ...
End
```

It is also possible to select a single value from a field by specifying the closest point

```
variable = <entity> <internal name> nearest location <X> <Y> <Z> as
 ↪<output name>
```

or a specified ID

```
variable = <entity> <internal name> at <entity> <ID> as <output name>
```

For example:

```
# Scope: Sierra > Procedure > Aria Region
Begin Heartbeat myHb
  ...
  Variable = Node solution->temperature nearest location 0.0 0.0 0.0 as␣
 ↪tempOrigin
  Variable = Node solution->temperature at node 123 as temp123
  variable = element volume_change_ratio at element 1 as vcr1
  ...
End
```

Note that this is similar to a *data probe*, with the key differences being that

1. The `nearest` specification snaps to the nearest node while the `data probe` interpolates to the exact point
2. The `data probe` is Eulerian (remains at a fixed point throughout the simulation), whereas the `nearest` specification is Lagrangian (sticks to a node and thus follows any deformation of the mesh at that node).

This distinction can be useful depending on what quantity you are interested in.

Note: Similar to results output, it is not fatal to request heartbeat of a global that does not exist. Instead, a warning is printed

```
*****************************************************************************
**  Warning:
**  In Heartbeat Output block 'THE_HEARTBEAT':
**  Global variable 'someMissingGlobal' does not exist and␣
 ↪will not be output
*****************************************************************************
```

The format of the heartbeat output is flexible, and is controlled by the parameters *Precision*, *Format*, *Labels*, and *Legend*. If no specific format is required, the following parameters

```
# Scope: Sierra > Procedure > Aria Region
Begin Heartbeat myHb
```

(continues on next page)

(continued from previous page)

```
  ...
  Timestamp Format = ""
  Precision        = 8   # precision of output
  Format           = csv # delimeter
  Labels           = Off # qnty=123 format
  Legend           = On  # header row of labels
End
```

produce a csv file that is human-readable and can be read easily across applications

```
          time, avg_temperature, max_temperature
0.00000000e+00,  3.50000000e+02,  4.00000000e+02
1.00000000e-01,  3.60000000e+02,  4.10000000e+02
2.00000000e-01,  3.70000000e+02,  4.20000000e+02
...
```

Output Scheduling

For both results and heartbeat output, the frequency of output is controlled by specifying either the time or step frequency using the following lines respectively

```
# Scope: Sierra > Procedure > Aria Region
Begin Heartbeat myHb
  at time 0.0 increment = 0.1
  # or
  at step 0 increment = 1
End
```

Note that multiple of these lines can be specified to define a more complex output schedule e.g.

```
# Scope: Sierra > Procedure > Aria Region
Begin Heartbeat myHb # or results output myOutput
  at time 0.0    increment = 100.0
  at time 300.0  increment = 300.0
  at time 1800.0 increment = 500.0
End
```

Additional times/steps can also be included using by specifying the respective command lines

```
# Scope: Sierra > Procedure > Aria Region
Begin Heartbeat myHb # or results output myOutput
  Additional Steps = 1 2 3
```

(continues on next page)

(continued from previous page)

```
  # or
  Additional Times = 0.1 0.2 0.3
End
```

By default, the time closest to the requested print interval is written. It is sometimes desired, however, to output exactly at a given print interval. In this case, `Timestep adjustment interval = <N>` can be set, which allows the time step manager to modify the next `N` steps to attempt to exactly hit the desired output.

```
# Scope: Sierra > Procedure > Aria Region
Begin Heartbeat myHb # or results output myOutput
  ...
  Timestep adjustment interval = 4
End
```

Finally, note that if a certain output pattern is desired to be used in multiple places, it can be defined as an *output scheduler* and given as input to the output block(s) in place of the arguments above. Note that the scheduler must be placed **at Sierra scope**:

```
Begin Sierra Job
  # Scope: Sierra!
  begin Output Scheduler mySched
    at time 0.0    increment = 100.0
    at time 300.0  increment = 300.0
    at time 1800.0 increment = 500.0
    ...
  end

  ...
  Begin Procedure theProcedure
      ...

      Begin Aria Region Aria_Region
        ...

        # Scope: Sierra > Procedure > Aria Region
        Begin Heartbeat myHb
          Use Output Scheduler mySched
          ...
        End

        Begin results output myOutput
          Use Output Scheduler mySched
```

(continues on next page)

(continued from previous page)

```
        ...
      End
    End Aria Region Aria_Region

  End Procedure theProcedure
End Sierra Job
```

Note: In the case of steady solutions, Aria output will contain two "time" steps: an initialization output plane, and the steady solution output. Care must be taken to use the appropriate time if these results are used in subsequent analysis (e.g. to provide the initial condition from file for a transient problem).

Restart Reference

Restart is used in analyses that may not complete in a single job submission and might complete when continued from a previous termination time. Aria maintains the ability to restart a previous analysis by recording the internal state variables of a problem in a restart database. In this case the input mesh is supplied from a *Finite Element Model* block and the restart information is obtained from the restart data. See the *command reference* for a detailed description of the lines possible when defining this block. See *Restart Examples* for examples using restart.

Note: Often one wishes to initialize an analysis with results from another analysis (e.g. a steady simulation fed into a transient IC). In these cases, a full restart is not necessary. Instead *Ic Input_Output_Region* and *Ic Read_File* can be used to initialize specific fields of interest using the *results* of the other simulation.

If one believes that a simulation might run too long, and wishes to record restart databases, they need only to add a *restart data* block as follows:

```
# Scope: Sierra > Procedure > Aria Region
Begin restart data data
  at step 0, increment = 1
  database name = restart.e
End
```

The restart database uses the same *output scheduling logic* as the results/heartbeat output, and can similarly be given an *output scheduler* block in place of specifying the commands in the restart block.

To restart a simulation, one must specify either the *restart time* to start at, or enable *automatic*

restart (latest available time, start of simulation if not found) **at Sierra scope**. Aria will read the given database name for the required restart information

```
Begin Sierra Job
  # Scope: Sierra!
  RESTART TIME = 0.00125
  # or: RESTART = auto

  ...
  Begin Procedure theProcedure
      ...

      Begin Aria Region Aria_Region
        ...

        # Scope: Sierra > Procedure > Aria Region
        Begin restart data data
          ...

          # Pointing to a previously created restart database
          database name = restart.e
        End
      End Aria Region Aria_Region

    End Procedure theProcedure
End Sierra Job
```

Warning: Specifying an **input database name** that does not exist would skip restart in previous versions, but this behavior is deprecated and moving forward will be treated as an error. To recover this behavior, one should specify only a **database name** line which will be used as both the input and output database. Alternatively, one can use an Aprepro switch to switch between reading and writing a restart database e.g.

```
# Scope: Sierra!
# is_restart = {is_restart = 0} # default to off
{if(is_restart)}
RESTART = auto
{endif}
```

```
# Scope: Sierra > Procedure > Aria Region
Begin restart data data
  ...

  {if(is_restart)}
    input database name = restart.e
  {else}
    output database name = restart.e
  {endif}
End
```

The same input can then be used by adding an Aprepro definition to the restart launch line

```
#First run
launch -n 8 aria -i aria.i

#Restart run
launch -n 8 aria -i aria.i --define "is_restart=1"
```

To save space, options are provided to minimize the retained restart data. These options include

- *Cycle Count* - Cycle timesteps within a single DB file
- *File Cycle Count* - Cycle timesteps within individual DB files
- *Overlay Count* - Overwrite intermediate time steps within a single DB file

IO Region

The Input_Output Region (IO Region) lets you use data from a previously-generated Exodus dataset (e.g. output from some other simulation code) in your simulation (as a source term, material property, initial condition, boundary condition, etc. . .).

It uses *transfer* and *solution control* to map the variables from the mesh they are saved on to the current problem's mesh.

Note: Just like other Sierra Regions the Input_Output Region must have its own Finite Element model command block defined. A full description of the syntax can be found in the *command reference*.

As an example, let us assume that an input mesh for an Input_Output Region contains a nodal variable `ConvCoeff` that we wish to use in another Region. In this case an outline for one-way transfer of `ConvCoeff` to a Region, `second_region`, in a steady-state problem would be:

```
Begin Sierra
  Begin Finite Element Model input_transfer
    ...
  End

  Begin Transfer my_first_transfer
    transfer commands for input_output_region to second_region
    SEND field hNd state none TO ConvCoeff state none
    ...
  End

  Begin Procedure My_Aria_Procedure
    Begin Solution Control Description
      Use System Main
      Begin System Main
        Begin Sequential MySolveBlock
          Advance io_region
          transfer my_first_transfer
          Advance second_Region
        End
      End
    End

    Begin Input_Output io_region
      USE FINITE ELEMENT MODEL my_input_transfer
    End
```

(continues on next page)

(continued from previous page)

```
    Begin Aria Region second_region
      use Finite Element Model input_transfer
      USER FIELD REAL NODE SCALAR ConvCoeff on surface_1
      ...
    End
    ...
  End
  ...
End Sierra
```

An example of an Input_Output Region used in an initial condition can be found *here*.

Controlling Time

The Input_Output Region can also be used with transient data. When the IO region contains transient data, there are several options for controlling which time value to transfer to the solution region. The IO region will interpolate linearly between its time steps when sending data to the solution region.

- **Fixed Time**: If you specify a fixed time in the IO region block, all transfers will use that time from the IO region data set. If the time you specify is larger than the last time in the IO region data set, the last time step will be used.
- **Offset Time**: This is a constant offset added to the simulation time. For example, if an offset of 10 is provided, at simulation time 0 the IO region will send its data at 10 seconds. The offset is added to the simulation time regardless of whether periodicity is active or not.
- **Start Time**: The start time should be provided along with `periodicity time`. If the start time is given as 25 and the periodicity as 10 then the simulation will use the time values between 25 and 35 from the IO region repeatedly once the simulation reaches 25. The default start time is 0.
- **Periodicity Time**: This controls how long of a time period to repeat. The default periodicity is 0 (not enabled).

The syntax for these commands are described in the *command reference section*.

When `fixed time` is not specified, the IO Region time is selected using the following logic:

```
if simulation_time > start_time and periodicity > 0
   io_region_time = start_time + (simulation_time - start_time) %␣
↪periodicity + offset
else
   io_region_time = simulation_time + offset
```

When `fixed time` was specified, `io_region_time = fixed_time`

Transfer

Transfer allows you to send fields from one region to another. There are two kinds of transfer - copy and interpolate. Copy transfer requires identical meshes (same node/element locations and same global IDs), while interpolation interpolates fields on dissimilar meshes. The benefit of the copy transfer is that a search is not required to transfer the information. With interpolation on the other hand, the sender and receiver need not be of same topology but the source and target destinations should overlap geometrically so that a search can be performed to find entities to interpolate.

The specification of this choice is as follows

```
Begin Transfer Transfer_name
  [TRANSFER_TYPE] [PART_TYPE] [TOPOLOGY] From [SRC_REGION] To [DST_REGION]
  ...
End
```

Here `SRC_REGION/DST_REGION` are the participating regions, `TRANSFER_TYPE` can be `COPY` or `INTERPOLATE`, `PART_TYPE` can be `SURFACE` or `VOLUME`, and `TOPOLOGY` can be `CONSTRAINTS`, `ELEMENTS`, or `NODES`. The remainder of this section discusses the major components that must be present in a Transfer block. See the *command reference* for a detailed list of all possible options.

Field

First, one must specify which quantities are to be transferred across regions. Each field transfer will have the general form

```
Send Field [SRC_FIELD] State [SRC_STATE] To [DST_FIELD] State [DST_STATE]
```

The `SRC_FIELD/DST_FIELD` can be any Aria field, including *Aria Solution Fields*, *User Fields*, and *Postprocess Value Fields*. Note that when receiving a solution field, the corresponding equation must be specified with a type of `XFER` so that the DOF field is created appropriately (see *Equation Specification*). See the *command reference* for more details on specifying a send field.

The `SRC_STATE/DST_STATE` refer to the state of the field (or `NONE` if the field is not stated). See *Internal Field Definition* for more information on field states.

For example, to transfer a postprocessed quantity in one region to a user field in another, one could do

```
# Scope: Sierra > Procedure
Begin Aria Region first_region
```

(continues on next page)

(continued from previous page)

```
  use finite element model same_model

  ...

  postprocess value of function "temperature - 273.15" on all_blocks as␣
↪tempC
End

Begin Aria Region second_region
  use finite element model same_model

  ...

  user field real node scalar first_temp on all_blocks
End

Begin Transfer temp_transfer
  copy volume nodes from first_region to second_region
  send field tempC state none to first_temp state none
End
```

Location

The source and destination parts dictate what portion of the domain is transferred. These parts be specified using the *Send Blocks* line command. Note that for copy transfer, each part must be specified in unique *send block* lines

```
# Scope: Sierra > Procedure
Begin Transfer foo
  Send Block block_1 to block_1
  Send Block block_2 to block_2
  ...
End
```

For interpolation transfer on the other hand, they must be specified in one line

```
# Scope: Sierra > Procedure
Begin Transfer foo
  Send Block block_1 block_2 to block_3 block_4
  ...
End
```

Note that in either case, assemblies can be used to simplify the definitions. Alternatively, the

transfer send and receive blocks can be specified in block form using *Send Blocks* and *Receive Blocks* respectively. This form is useful when one wishes to send or receive all blocks in the mesh, since it allows the use of the `Include All Blocks` alias instead of having to list them manually.

```
# Scope: Sierra > Procedure
Begin Transfer foo
  Begin Send Blocks
    Include All Blocks
  End
  Begin Receive Blocks
    Include Block = block_3 block_4
  End
  ...
End
```

Note that if these lines are omitted, the transfer is performed over the entire domain.

Search Tolerance

In cases of interpolation, a search over the two meshes must be performed to communicate values. By default, the tolerance used in this search is a function of the geometric problem size. For cases in which the source and destination do not overlap, one should specify tolerancing in order to guarantee that values will be mapped to the destination. If the the separation between source and destination varies, then one should specify a *Geometric Tolerance* consistent with the largest separation distance.

Extrapolation Handling

When source and destination entities partially overlap, then by default source values will be extrapolated to the destination. This behavior can be explicitly controlled with the *Nodes Outside Region* command. The most common options for the command are described below:

EXTRAPOLATE
: This is the default behavior. The sending field is extrapolated beyond the bounds of the sending mesh. This can lead to extrapolation error, such as when a large gradient at the surface causes a negative values when only positive values are acceptable. If this happens, upper and lower bounds can be set per field within the `SEND FIELD` command.

```
Send Field field1 State none To field2 State none Lower Bound 0.0␣
↪Upper Bound 1.0
```

TRUNCATE
: The receiving coordinate is projected back to the surface of the sending mesh to determine a

value. This ensures that the receiving value is not outside of the field values in the sending mesh.

ABORT
: If any receiving point is outside the sending mesh by more than the geometric tolerance, abort the simulation. Do not attempt to project, extrapolate, or otherwise handle the point. This can be useful for detecting problem setup errors that otherwise may have been silently extrapolated.

IGNORE
: The receiving mesh object can be ignored and will receive no value. This is usually not recommended as it can cause mesh objects just outside to have a zero value when the nodes just inside the mesh might have very large values.

Skeleton Examples

Since several different uses of transfer can arise, some examples are included below.

Note: While this example shows only steady problems, the same basic setup of transfer would apply to transient problems as well.

A skeleton outline of one-way transfer between two regions

```
# Scope: Sierra > Procedure
Begin Solution Control Description
  Use System Main
  Begin System Main
    Begin Sequential MySolveBlock
      Advance first_Region
      transfer my_transfer
      Advance second_Region
    End
  End
  ...
End

Begin Aria Region first_region
  Use Finite Element Model fem_model_A

  eq energy for temperature On block_1 using q1 with lumped_mass diff
  ...
End
```

(continues on next page)

(continued from previous page)

```
Begin Aria Region second_region
  Use Finite Element Model fem_model_B

  eq energy for temperature On block_1 using q1 with xfer
  ...
End

Begin Transfer my_transfer
  interpolate volume nodes from first_region to second_region
  send block block_1 to block_1
  send field solution->temperature state new to solution->temperature␣
↪state new
End
```

A skeleton outline of two-way transfer between two regions:

```
# Scope: Sierra > Procedure
Begin Solution Control Description
  Use System Main
  Begin System Main
    Begin Sequential MySolveBlock
      Advance RegionA
      Transfer AtoB
      Advance RegionB
      Transfer BtoA
    End
  End
  ...
End

Begin Aria Region RegionA
  Use Finite Element Model same_model

  eq energy for temperature On block_1 using q1 with diff
  eq species for species of 3 On block_2 using q1 with xfer
  ...
End

Begin Aria Region RegionB
  Use Finite Element Model same_model

  eq energy for temperature On block_1 using q1 with xfer
  eq species for species of 3 On block_2 using q1 with diff
```

(continues on next page)

(continued from previous page)

```
  ...
End

Begin Transfer AtoB
  copy volume nodes from first_region to second_region
  send block block_1 to block_1
  send field solution->temperature state new to solution->temperature␣
↪state new
End

Begin Transfer BtoA
  copy volume nodes from second_region to first_region
  send block block_2 to block_2
  send field solution->species_3 state new to solution->species_3 state␣
↪new
End
```

Transfer is not exclusive to Aria regions. For example, an *IO Region* can be used to distribute data from external mesh databases. See *Using Transfer* for a complete example of this use case.

General Field Transfers

Fields can be stored at nodes, elements, or faces. Below are common examples of transfer blocks for field transfers wherein the fields are stored on different mesh entities (e.g., nodal field vs element field).

To send an element field to a nodal field via a constant approximation:

```
Begin Transfer ElementToNodeConstantApproximation
  interpolate volume elements from first_region to second_region
  from elements to nodes
  interpolation function element_centroid_constant
  send field MY_ELEMENT_FIELD state new to MY_NODAL_FIELD state new
End
```

To send an element field to a nodal field via a linear least squares approximation:

```
Begin Transfer ElementToNodeLinearApproximation
  interpolate volume elements from first_region to second_region
  from elements to nodes
  interpolation function element_centroid_linear
  send field MY_ELEMENT_FIELD new to MY_NODAL_FIELD state new
End
```

By default, `element_centroid_linear` equates to a linear least squares approximation. Alternative approximations are available via `patch recovery evaluation = LINEAR LEAST SQUARES|LINEAR MOVING LEAST SQUARES|QUADRATIC LEAST SQUARES|QUADRATIC MOVING_LEAST_SQUARES`. Patch recovery, as the name implies, forms the specified approximation from the surrounding patch of elements.

To send an element field (e.g., shell elements) to a face field:

```
Begin Transfer ElementToFaceConstantApproximation
  interpolate volume elements from first_region to second_region

  Begin Send Blocks
    Include All Blocks
  End
  Begin Receive Blocks
    Include Block = surface_1
  End

  from elements to faces
  interpolation function element_centroid_constant
  send field MY_ELEMENT_FIELD state none to MY_FACE_FIELD state none
End
```

4.10 Coupling

File Based Coupling

In a file-based coupling, information is exchanged across apps via results files. This typically implies running one app, then using those results to run the other app. In a one-way coupling that is the end of the simulation, otherwise this process can be cycled until some user-determined convergence metric is met. Within a single simulation, the data from the coupled simulation(s) are typically incorporated by *transfering* from an *IO Region*. The IO region handles interpolating data to the current simulation time and mesh. The simulation results can write to an output database to feed into the coupled application.

Note: A file-based coupling is useful for time-varying data generated from an external app. If it is only necessary to apply initial conditions from an external source (e.g. a steady simulation fed into a transient IC), the *Ic Input_Output_Region* and/or *Ic Read_File* ICs can be a much simpler alternative.

A file-based coupling input deck will look similar to the example below

```
Begin Sierra sierra

  #! 1) The domain Aria solves
  Begin Finite Element Model The_Mesh
    Database Name = mesh.g
    Decomposition Method = rcb

    use material foo for block_1
    use material bar for block_2
    ...
  End

  #! 2) The coupled domain (includes results)
  Begin Finite Element Model The_Coupled_Subdomain
    Database Name = coupling_surf.e
    Decomposition Method = rcb
  End

  ...
  Begin Procedure proc
    Begin Solution Control Description
      Use system main
      Begin system main
        #! 3) Define how a step is taken
        Begin Transient my_transient
          Advance coupled_app_region
          Transfer coupled_to_aria
          Advance aria_region
        End
        ...
      End
    End

    ...

    Begin Aria Region aria_region
      #! 1) The entire Aria domain
      Use Finite Element Model The_Mesh

      #! 4) Set up a user field to be used (also created expressions)
      user field real node scalar my_coupled_qnt on coupled_surface

      ...
```

(continues on next page)

(continued from previous page)

```
    #! 5) Write out the information needed by coupled app
    Begin Results Output out
      database name = to_coupled_app.e
      at step 0, increment = 1
      nodal variables = solution->temperature as T
      ...
    End
  End

  #! 6) Create an IO region to manage coupled interface
  Begin Input_Output Region coupled_app_region
    Use Finite Element Model The_Coupled_Subdomain
  End

  #! 4) Transfer relevant information from couple app to Aria
  Begin transfer coupled_to_aria
    interpolate volume nodes from coupled_app_region to aria_region
    Send block surface_1 to coupled_surface
    Send Field my_coupled_qnt State None to my_coupled_qnt State none
    ...
  End
End Procedure
End Sierra
```

1. Just like a typical simulation, one must specify the *finite element model* defining the Aria domain
2. Typically, the coupling occurs only on a subset of the domain. Rather than interpolate over the entire domain, one can specify only the subdomain of interest as the finite element model of the Input_Output region. In any case, this database should contain not only the mesh information of the coupled interface, but also the relevant results needed to run the Aria simulation.
3. As part of the *transient*, the IO region must be advanced so that the appropriate information can be transferred to the aria region.
4. Set up the information *transfer* from the IO region to the appropriate Aria region fields. The receiving field on the Aria side is typically a DOF field (defined as an *XFER EQ*) or a *user field* which can then be used in the Aria expression system.
5. In order to provide the results of this simulation to the coupled app, write an *output database*.
6. Create the IO region with the data and geometry specified in 2.

Sierra Coupling

Unlike a file-based coupling, Sierra coupling means that information is being exchanged *during* simulation. In general, it will be fairly similar syntactically to a *File Based Coupling* with the exception that the information comes not from an IO Region but rather other Sierra app regions (still using *Transfer* between them).

Examples of this include:

- Arpeggio/ Arpeggitar for coupling with Adagio.
- `fuego_aria` for coupling with Fuego.

MPMD Coupling

Similar to a *Sierra Coupling*, MPMD coupling exchanges information *during* simulation. The key difference however, is that information is exchange not via Sierra framework transfers, but rather external coupled apps using MPMD defined in the *Mpmd Coupling* block. For more details on MPMD couplings with individual applications, see the respective coupling manuals:

- *PMR*
- Sparc-Aria

For information on launching an MPMD simulation, see *Running MPMD Jobs*.

5 Tutorials & Examples

The present section offers references for setting up your own case(s). First the following list is a set of tutorials (many coming from the Sierra100 courses). These tutorials offer step-by-step directions for setting up a simple case

5.1 Sierra 100

Basic Heat Conduction

This tutorial shows how to set up a basic transient heat conduction problem in Aria. This module has also been recorded in prior trainings and is available here or in the player below.

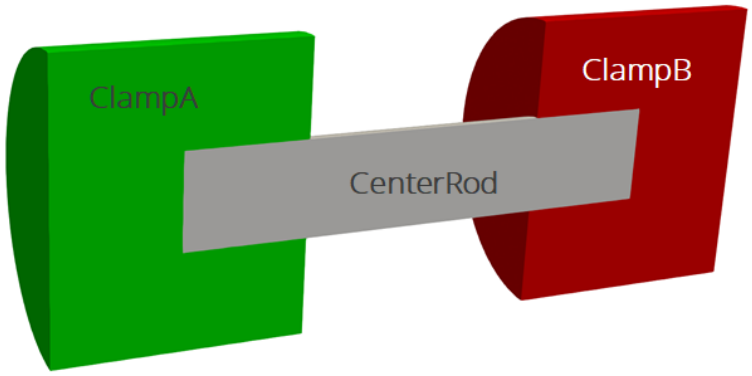
Problem Files

The files required for this tutorial can be downloaded here, or found at `$SIERRA100/TF/Aria` (where `SIERRA100=/projects/sierra100` in a CEE environment, or your Sierra test distribution otherwise).

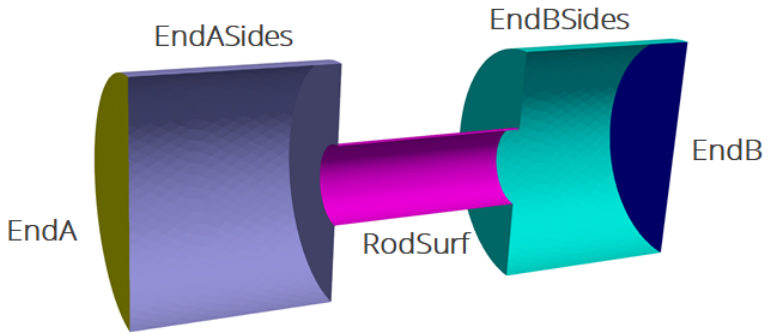
- `Aria input file`
- `Mesh file`
- `Training slides`

Problem Domain/Mesh

The geometry we will use for this demonstration consists of three cylinders making a dumbbell shape, with three separate named blocks (ClampA, ClampB, and CenterRod). Blocks (volumetric sections in the mesh) are used in Aria to define different material properties, and can also be used to run post-processors on.



The problem also includes five separate named sidesets (EndA, EndB, EndASides, EndBSides, and RodSurf) applied on different surfaces of the mesh. Sidesets in aria are used to apply boundary conditions or do surface post-processing. Any surface not covered by a sideset will be treated as adiabatic, as will any sideset with no boundary condition applied to it.



The mesh file needed for this tutorial can be downloaded here, or it can be generated manually using the Cubit journal file below.

Cubit Journal File

```
undo on

create cylinder height 0.1 radius 0.01
create cylinder height 0.05 radius 0.03
create cylinder height 0.05 radius 0.03
volume 2 move z 0.05
volume 3 move z -0.05
subtract volume 1 from volume 2 3 keep
delete volume 2 3
merge all
imprint all
merge all

volume 1 size 0.003
volume 4 5 size 0.005

volume all tetmesh growth_factor 1.05
volume all scheme tetmesh
mesh volume all

block 1 add volume 1
block 1 name "CenterRod"
block 2 add volume 4
block 2 name "ClampA"
block 3 add volume 5
block 3 name "ClampB"


sideset 1 add surface 14
sideset 1 name "EndA"

sideset 2 add surface 18
sideset 2 name "EndB"

sideset 3 add surface 20
sideset 3 name "RodSurf"

sideset 4 add surface 13 12
sideset 4 name "EndASides"
```

(continues on next page)

(continued from previous page)

```
sideset 5 add surface 17 19
sideset 5 name "EndBSides"

undo group begin
set exodus netcdf4 off
set large exodus file on
export mesh "mesh.g" overwrite
undo group end
```

Aria Input File

Every Aria simulation requires a text input file to define the problem. The following sections show how to set up that file to run a transient heat conduction problem using the geometry shown above.

Indentation is recommended but not required in an Aria input file. You can indent with tabs or spaces, or choose not to indent things at all - although we highly recommend using indentation (with spaces) to make a consistently readable input file and to avoid errors with commands going in the wrong scope.

Anything that starts with a # or a $ will be treated as a comment and ignored. Long lines can be split using \$, like this

```
# This is a comment
$ and so is this

# This is a single line command split on three lines
List of Blocks = block_1 block_2 block_3 \$
                 block_4 block_5 block_6 \$
                 block_7 block_8
```

Overall File Structure

The Aria input file is structured using nested blocks with `Begin` and `End` commands surrounding each block. The outermost block must always be the SIERRA block. Commands directly in this block are referred to as "Domain-level" commands. Other commands must go inside inner blocks, such as "Procedure" or "Aria Region", and will be referred to as such. If you put a command in the wrong level, you will get an error when you try to run Aria.

```
Begin SIERRA Aria
  # Domain-level commands
```

(continues on next page)

(continued from previous page)

```
  Begin Procedure AriaProcedure
    # Procedure-level commands

    Begin Solution Control Description
      # Solution control commands
    End

    Begin Aria Region myRegion
      # Region-level commands
    End
  End
End
```

Defining Materials

Scope: Domain

You must define a material to use for each block in your mesh. A material can be used on multiple blocks, but every block must have a material. The properties required for your material will depend on what equations you are solving. (For more information on material definition refer to *Material Properties*).

A transient heat conduction problem like this requires four properties:

- Density
- Specific heat
- Thermal Conductivity
- A model for heat flux

The syntax for setting up a material block named "aluminum" with constant properties is shown below:

```
Begin Aria Material aluminum
    Density              = Constant value = 2770  # kg/m^3
    Specific Heat        = Constant value = 800.0 # J/kg-K
    Thermal Conductivity = Constant value = 175.0 # W/mK
    Heat Conduction      = Generalized
End   Aria Material aluminum
```

The general format for specifying a material property is:

```
PROPERTY = MODEL [MODEL ARGS]
```

Some common examples of different models are Constant, Polynomial, Scalar_String_Function, and many others. Each of these models requires a different set of arguments. Examples of how to use these are shown below.

```
Density               =  Constant               value = 100
Specific Heat         =  Polynomial             order=1 C0=1 C1=1␣
↪variable=density
Thermal conductivity  =  User_Function          name = my_fcn X = time
Viscosity             =  Scalar_String_Function f = "2*temperature"
```

For a complete list of the available models to use for a given property, you can refer to the *Material Properties* section of the user manual. For example, to see all available models for density you could go to the *density section*.

Define the Finite Element Model

Scope: Domain

The finite element model section is where you indicate which mesh file to use and associate materials with the blocks and surfaces in the mesh.

```
Begin Finite Element Model FEModel
    Database Name = mesh.g
    Use Material aluminum for ClampA ClampB CenterRod
End   Finite Element Model FEModel
```

Linear Solver

Scope: Domain

Every problem requires at least one linear solver to use to solve the generated linear system. Aria includes several preset solvers that can be used for common problem types. For this example we will use the "Thermal_Symmetric" preset, which uses the CG solver with a Jacobi preconditioner.

```
Begin tpetra equation solver solve_temperature
  Begin Preset Solver
    Solver Type = Thermal_Symmetric
  End
End
```

Solution Control

Scope: Procedure

The solution control block defines how the Aria region gets executed - a single time for a steady-state problem or multiple times for a transient problem. When the problem is transient, this is also where you specify the time step selection method, time integration scheme, and other parameters related to time step selection.

```
$-----------------------------------------------------
$       Define temporal solution parameters
$-----------------------------------------------------
    Begin Solution Control Description
        Use System Main

        Begin System Main
            Begin Transient Solution_block_1
                Advance AriaRegion
            End
        End

$-----------------------------------------------------
$           Specify time integration settings
$-----------------------------------------------------
        Begin Parameters for Transient Solution_Block_1
            Start Time       = 0.0   # seconds
            Termination Time = 600.0 # seconds

            Begin Parameters for Aria Region AriaRegion
                Time Step Variation = Adaptive
                Initial Time Step Size = 0.05
                Time Integration Method = BDF2
                Maximum Time Step Size = 20.0
                Minimum Time Step Size = 0.01
                Predictor-Corrector Tolerance = 0.01
            End
        End
    End
```

Aria Region

Scope: Procedure

The Aria Region is where you specify nonlinear solution settings, what equation you are solving, boundary conditions, initial conditions, postprocessors, and output controls.

Before getting to the problem-specific selections, there are a few general settings required in the Region block:

- Which linear solver to use
- Nonlinear solution settings (strategy, tolerances, etc)
- Which finite element model block (mesh) to use. Each region must be associated with exactly one mesh.

The following block shows a standard set of parameters, which we will use for this example.

```
Begin Aria Region AriaRegion

  $-----------------------------------------------
  $ Define linear and nonlinear solver parameters
  $-----------------------------------------------
  Use Linear Solver solve_temperature
  Nonlinear Solution Strategy  = Newton
  Maximum Nonlinear Iterations = 10
  Nonlinear Residual Tolerance = 1.e-6

  $-------------------------------------------------------
  $ Specify which mesh model to use for this region
  $-------------------------------------------------------
  Use Finite Element Model FEModel

  # other region commands
End
```

Governing Equation

Scope: Aria Region

The equations to solve are specified with one or more "EQ" lines. The important specifications on the EQ line are the equation name (Energy), the degree-of-freedom (DOF) name (Temperature), the location to solve the equation on (all_blocks), and the terms to activate in the equation (Mass and Diff).

```
#  NAME       DOF             LOCATION                  TERMS
EQ Energy for Temperature on all_blocks using Q1 with Mass Diff
```

This generates a governing equation of

$$\rho C_p \frac{\partial T}{\partial t} + \nabla \cdot \vec{F} = 0$$

The "mass" term is required for transient problems, and adds the $\rho C_p \frac{\partial T}{\partial t}$ term, and the "diff" term adds the $\nabla \cdot \vec{F}$ term.

When we set up our *material properies* we defined that $\vec{F} = -k\nabla T$ when we set the heat conduction model as "Generalized", so the overall governing equation we are solving is

$$\rho C_p \frac{\partial T}{\partial t} + \nabla \cdot (-k\nabla T) = 0$$

Initial Conditions

Scope: Aria Region

Since this is a transient problem, initial conditions are required for all DOFs on all blocks. The syntax for setting initial conditions is very similar to the syntax used to set material properties

```
IC for DOF on BLOCK = MODEL [MODEL ARGS]
```

To set a constant initial condition for temperature of 300 K, we would use

```
IC for Temperature on all_blocks = constant value = 300 # K
```

To see what other models are available to use for initial conditions you can refer to the *initial condition section* in the user manual - common models include constant, user_function, and scalar_string_function.

Boundary Conditions

Scope: Aria Region

As with initial conditions and material properties, boundary conditions can be specified by selecting a model and providing its arguments in-line. There are two primary categories of boundary conditions in Aria - Dirichlet (which specify the DOF on the boundary) and Flux (which specify flux of the conserved quantity). The syntax for these is

```
BC Dirichlet for DOF on SIDESET = MODEL [MODEL ARGS]
BC Flux for EQUATION on SIDESET = MODEL [MODEL ARGS]
```

For this problem we are going to set Dirichlet (fixed temperature) boundary conditions on the ends of the clamps (EndA and EndB). To make the problem have some time variation, we will use a scalar string function to define a time-varying condition on one end of the domain.

```
BC Dirichlet for temperature on EndA = constant value = 300 # K
BC Dirichlet for temperature on EndB = scalar_string_function f =
 ↪"350+100*sin(t/300)" # K
```

To see all the available boundary condition models, you can use the *boundary condition section* in the user manual.

Solution Output

Scope: Aria Region

To output our results, we need to define a Results Output block, which will define the name of the file we are writing to, which fields to write, and how often to write. For the example below, we will write the temperature solution field to a variable called "T" in heat_cond.e every 5 time steps. Output can also be specified at a given time interval instead of a step interval using `At time 0 Increment = 0.1`.

```
Begin Results Output AriaOutput
    Title Aria: Transient Training Model
    Database Name = heat_cond.e
    At Step 0 Interval = 5
    Nodal Variables = Solution->Temperature as T
End
```

Running Aria

Once you have finished setting up the input file, you are ready to run Aria. From a CEE unix environment, you can load the sierra module to access the latest release of Aria to run it on your local machine. To run it on 4 processors, you could use

```
$ module load sierra
$ launch -n 4 aria -i heat_cond_session1.i
```

Reading the Log File

As it runs, Aria will output continuously to a log file, by default with the same base name as your input file (e.g. "heat_cond_session1.log"). You can check the log file to monitor the job progress and to check that it is adequately solving your specified equations. The log file is also where you should look if the solution terminates with an error.

```
Transient solution_block_1: dt = 0.05
Transient Solution_Block_1, step 1, time 5.0000e-02, time step 5.0000e-02,
 ↪ 0.00% complete
Memory Usage: current = 302419968 (288.4 M), high-water-mark = 421240832␣
 ↪(401.7 M)
DEFAULTING PREDICTOR-CORRECTOR_BEGIN_AFTER_STEP = 3.

Equation System AriaRegion->main:
* Step  : Transient, Strategy: NEWTON, Time: 5.00e-02, Step: 5.00e-02
* Matrix: Solver: "solve_temperature", Unknowns: 6122, Nonzeros: 83476
* Mesh  : Processor 0 of 1: 30689 of 30689 elems, 6122 of 6122 nodes
  N O N L I N E A R                 L I N E A R
----------------------   ---------------------------------------
Step  Resid    Delta     Itns  Status   Resid    Asm/Slv Time
---- -------- --------   ---- -------- -------- ---------------
  1 2.27e+03 5.80e+02     10       ok 4.07e-07 2.8e-02/2.6e-03
  2 9.25e-04 2.83e-04     14       ok 3.20e-07 2.8e-02/2.9e-03
  3 2.96e-10                     NoOp          2.7e-02
Termination reason: 2.95766e-10 < nonlinear_residual_tolerance(1e-06),
                and 0 < nonlinear_correction_tolerance(1e-06)
```

In the example log file output above, we can see that Aria took three nonlinear iterations to converge on the time step shown here, and that the final nonlinear residual was suitably low. The linear solver residuals and iteration counts are also shown here, and can be an indication of whether the solver you selected is struggling with the current problem (in this case it is not).

```
-------------------------------------------------
There were no errors encountered during parse
There was 1 warning encountered during parse
There were no errors encountered during execution
There were no warnings encountered during execution
+----+----+----+----+----+----+----+----+----+----+----+----+----+----+---
 ↪-+----
SIERRA execution successful.
    For region AriaRegion
      Number of timesteps            (failed) : 0
      Number of timesteps            ( total) : 190
```

(continues on next page)

(continued from previous page)

```
      Number of nonlinear iterations ( total) : 394
    Memory Usage: current = 311021568 (296.6 M), high-water-mark =␣
↪421838848 (402.3 M)
```

At the end of the log file there is a summary showing that the problem executed successfully and the number of time steps, failed time steps, and nonlinear iterations.

If you want to output other quantities than just the temperature, you can also check the log file header (near the top) for a list of all available fields (nodal, face, element, and global) that are available to output.

```
Aria Region "Aria_region" has the following fields available for output:
  NODE_RANK Fields:
    * physical_coordinates
    * pp->HEAT_CONDUCTION
    * predicted_solution->TEMPERATURE
    * solution->TEMPERATURE
    * time_derivative_at_time->TEMPERATURE
  FACE_RANK Fields:
  * thermal_contact_status
  ELEMENT_RANK Fields:
    * current_element_volume
    * volume_change_ratio
  Global Variables:
    * p_inner
```

Viewing the Results

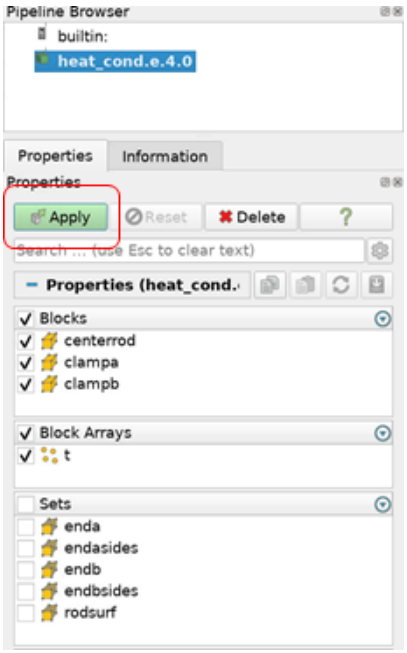
The two common tools used at Sandia for visualizing Exodus files are Ensight and Paraview. There are training courses available for both tools, so their use is not a focus of this tutorial. Examples shown here will use Paraview.

From a CEE unix environment with graphics, you can launch paraview using

```
$ module load viz
$ paraview heat_cond.e.4.0
```

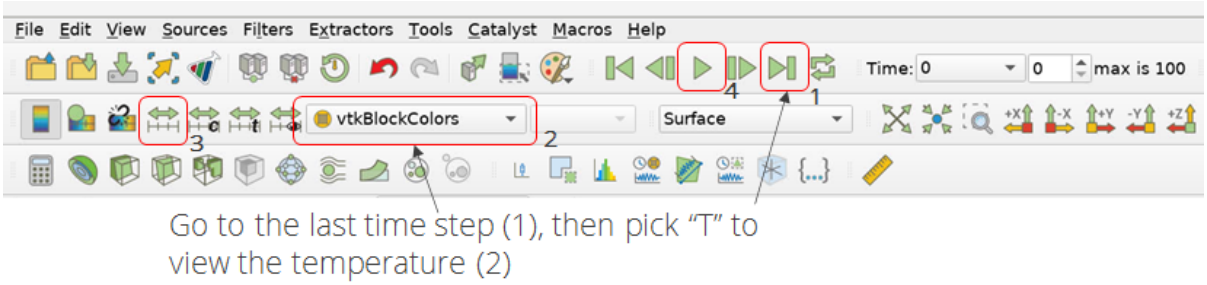
Or if you have installed Paraview locally, you can launch it and select the exodus file from the appropriate file opening menu.

Once you have loaded it, you need to select which variables to show then click “Apply”

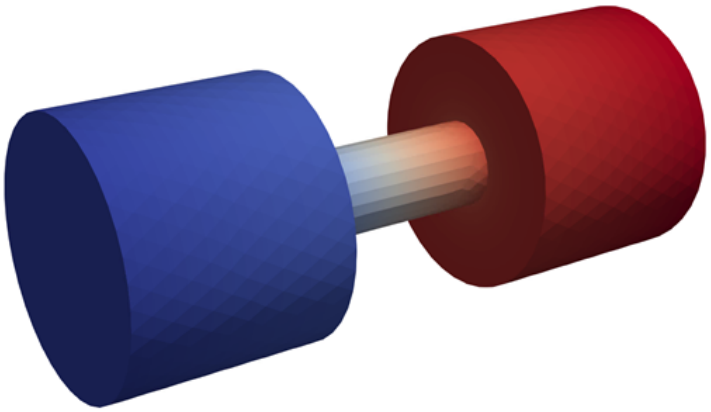


To show the temperature field on the object, you can

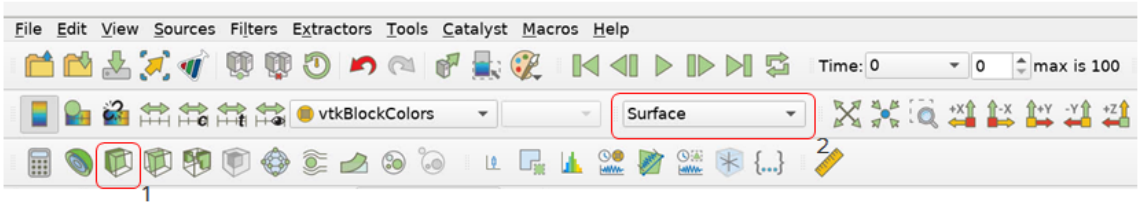
1. Go to the last time step
2. Select the temperature variable to show
3. If you showed temperature first the scale may be off. Rescale it.
4. Click the Play button to animate the results over time.



The result shows the temperature distribution is as-expected.

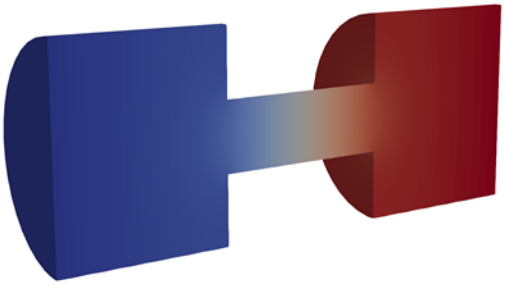


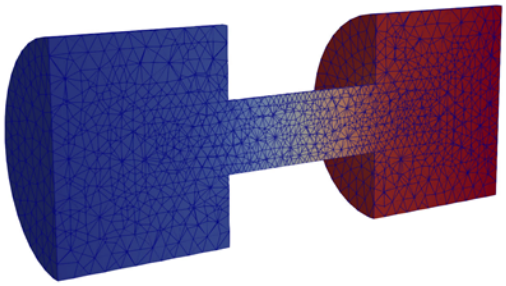
Next, we can slice the domain in half and look at the internal profile and the mesh details



Click the Clip filter (1) then "Apply" to cut the model in half

Select "Surface with Edges" (2) to view the mesh





Materials and Flux BCs

This tutorial builds on the prior *Basic Heat Conduction* tutorial and shows how to add a more complicated material and set flux boundary conditions. This module has also been recorded in prior trainings and is available here or in the player below.

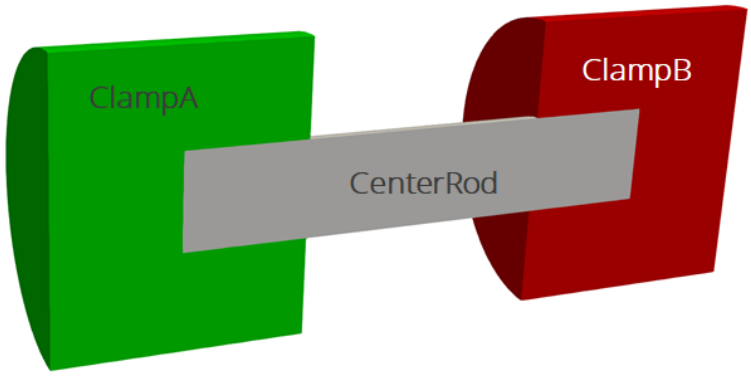
Problem Files

The files required for this tutorial can be downloaded here, or found at `$SIERRA100/TF/Aria` (where `SIERRA100=/projects/sierra100` in a CEE environment, or your Sierra test distribution otherwise).

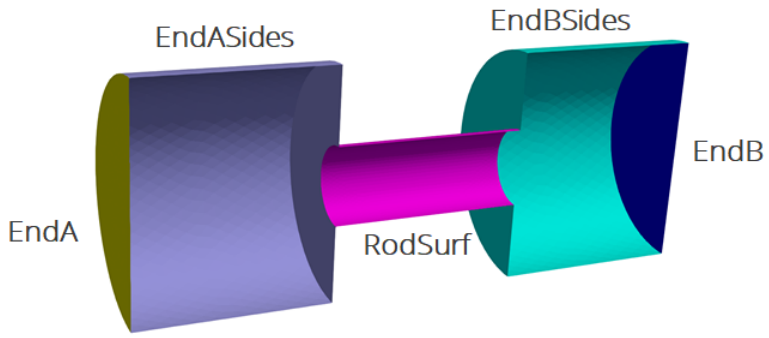
- `Aria input file`
- `Mesh file`
- `Training slides`

Problem Domain/Mesh

The geometry we will use for this tutorial is the same as the one used *Basic Heat Conduction*. It consists of three cylinders making a dumbbell shape, with three separate named blocks (ClampA, ClampB, and CenterRod). Blocks (volumetric sections in the mesh) are used in Aria to define different material properties, and can also be used to run post-processors on.



The problem also includes five separate named sidesets (EndA, EndB, EndASides, EndBSides, and RodSurf) applied on different surfaces of the mesh. Sidesets in aria are used to apply boundary conditions or do surface post-processing. Any surface not covered by a sideset will be treated as adiabatic, as will any sideset with no boundary condition applied to it.



The mesh file needed for this tutorial can be downloaded `here`, or it can be generated manually using the Cubit journal file shown in the *Basic Heat Conduction* tutorial.

Defining Complex Materials

Scope: Domain

In the prior tutorial we used a single material with constant properties for all blocks in the mesh. For this example we will add a second material - 304 stainless steel - and use it on one of the blocks. This material will use temperature-dependent properties for specific heat and thermal conductivity. For more information on material definition refer to *Material Properties*.

```
Begin Aria Material ss304
  Density              = Constant value = 8000
  Specific Heat        = Polynomial Variable=Temperature order=1 C0=500␣
↪C1=0.1
  Thermal Conductivity = User_Function name=ssteel_k_function␣
↪X=temperature
  Heat Conduction      = Generalized
End   Aria Material ss304

Begin Definition for Function ssteel_k_function
  Type is Piecewise Linear
  Begin Values
    #T (K)  k (W/mK)
    100     10
    500     20
  End Values
End   Definition for Function ssteel_k_function
```

We used a *user function* to define the thermal conductivity from a table of values at different temperatures. Since it is a `Piecewise Linear` table, the value for conductivity will be interpolated linearly between the points of the table.

We also used a polynomial model to define a linear specific heat ($C_p = 500 + 0.1T$) - we could

also have used a *string function* to get the same result

```
Specific Heat = Scalar_String_Function f = "500 + 0.1*temperature"
```

For a complete list of the available models to use for a given property, you can refer to the *Material Properties* section of the user manual. For example, to see all available models for specific heat you could go to the *specific heat section*.

Applying the new Material

Scope: Domain

To apply the new stainless steel material to one of the blocks, we must edit the `Finite Element Model` block where mesh blocks were previously all assigned to use the aluminum material.

```
Begin Finite Element Model FEModel
    Database Name = mesh.g
    Use Material aluminum for ClampA ClampB
    Use Material ss304 for CenterRod
End   Finite Element Model FEModel
```

Updating Boundary Conditions

Scope: Aria Region

In the prior tutorial we used Dirichlet boundary conditions - specified temperature - on both ends of the domain. For this case we will change one of those boundary conditions to a specified time-varying flux. Delete the Dirichlet BC from `EndB` and replace it with the following flux BC:

```
BC Flux for Energy on EndB = scalar_string_function f = "-5e4*sin(t/300)"
```

Note: The sign convention for flux BCs in Aria is that positive = outflux.

To see all the available boundary condition models, you can use the *boundary condition section* in the user manual.

Post-process Flux

Scope: Aria Region

Aria supports a large number of different *postprocessors*. Postprocessors can be used to calculate spatially varying fields of material properties or intermediate calculations, or can be used to do reductions (integral, average, min, max, etc) to produce global variables. In this case, we will use the `integrated_flux` postprocessor to calculate the total flux on both ends of the domain.

```
Postprocess integrated_flux of equation energy on EndA as FluxA
Postprocess integrated_flux of equation energy on EndB as FluxB
```

This will produce two global variables - `FluxA` and `FluxB`. These will be printed to the log file, and can also be output to a *heartbeat* file (a text file) or included in the regular *exodus output*. For this example we will just add them to the existing exodus output file.

```
Begin Results Output AriaOutput
    Title Aria: Transient Training Model
    Database Name = heat_cond.e
    At Step 0 Interval = 5
    Nodal Variables = Solution->Temperature as T

    Global Variables = FluxA
    Global Variables = FluxB
End
```

Running Aria

Once you have finished setting up the input file, you are ready to run Aria. From a CEE unix environment, you can load the sierra module to access the latest release of Aria to run it on your local machine. To run it on 4 processors, you could use

```
$ module load sierra
$ launch -n 4 aria -i heat_cond_session2.i
```

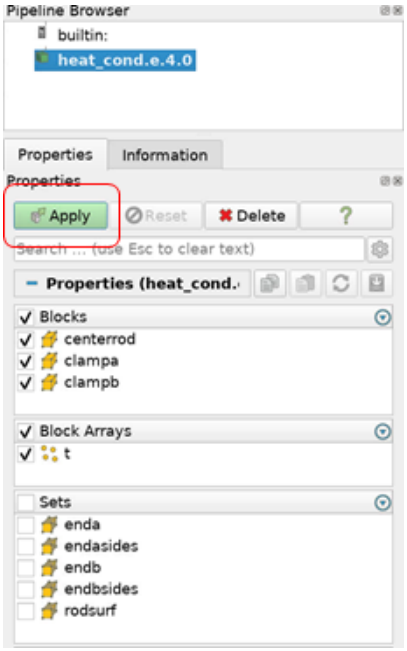
Viewing the Results

From a CEE unix environment with graphics, you can launch paraview using

```
$ module load viz
$ paraview heat_cond.e.4.0
```

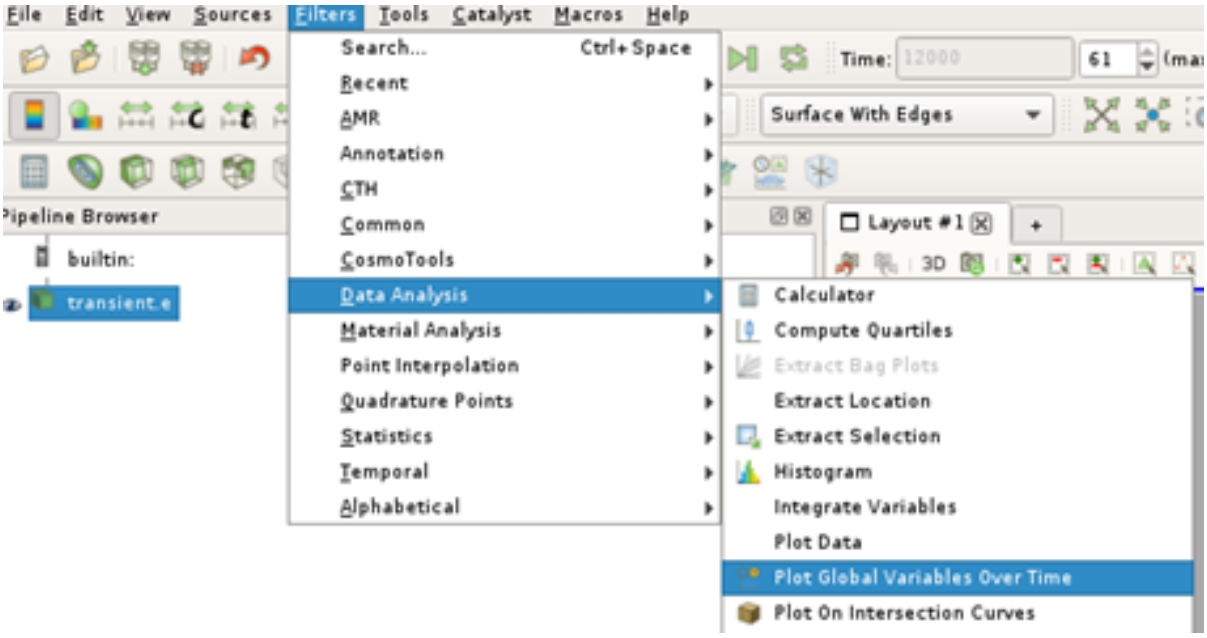
Or if you have installed Paraview locally, you can launch it and select the exodus file from the appropriate file opening menu.

Once you have loaded it, you need to select which variables to show then click “Apply”

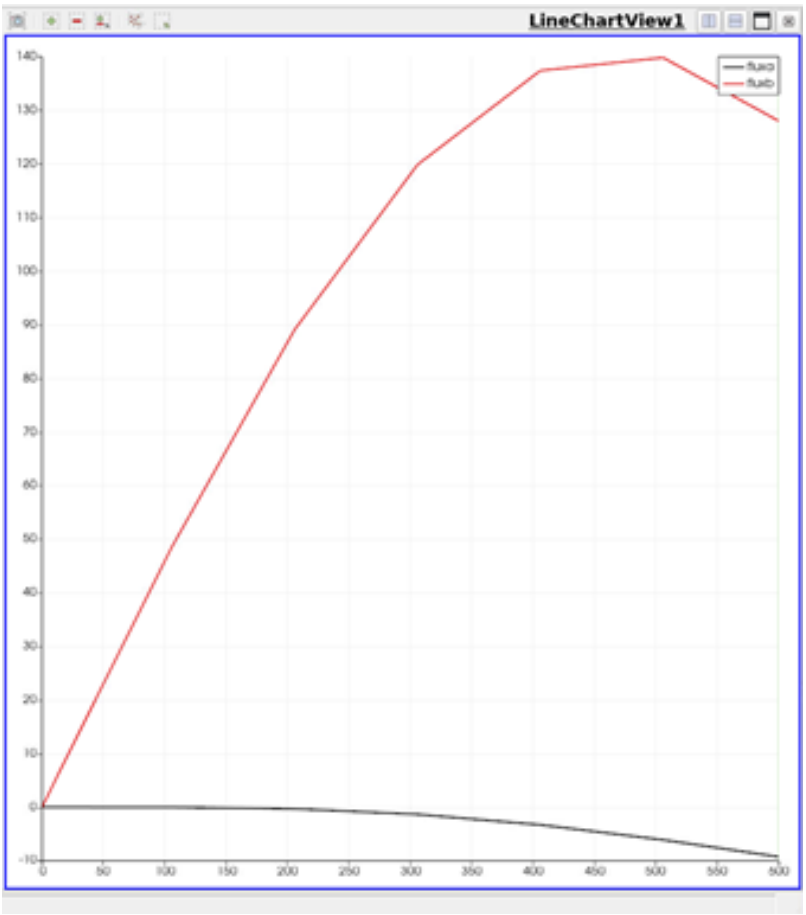


To plot the global variables (fluxes on the two ends of the domain)

1. Go to the Filters menu and select “Data Analysis” then “Plot Global Variables Over Time”
2. Click “Apply” to generate the plots



The result shows how the flux varies over time



Convective Flux BCs

This tutorial builds on the prior *Materials and Flux BCs* tutorial and shows how to add a convective flux boundary condition. This module has also been recorded in prior trainings and is available here or in the player below.

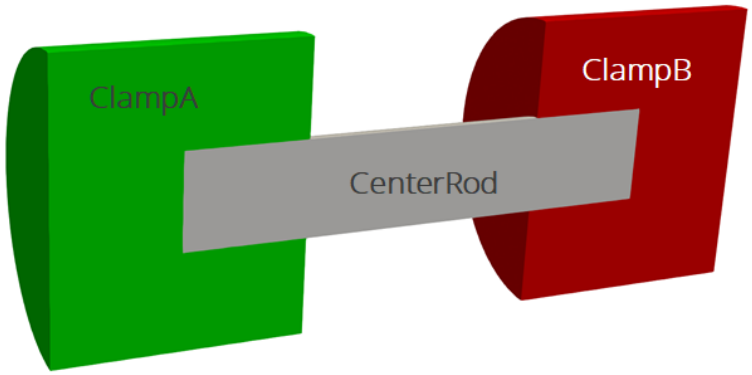
Problem Files

The files required for this tutorial can be downloaded here, or found at `$SIERRA100/TF/Aria` (where `SIERRA100=/projects/sierra100` in a CEE environment, or your Sierra test distribution otherwise).

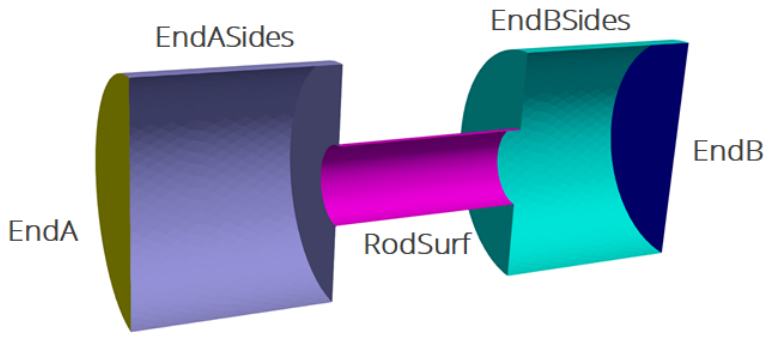
- `Aria input file`
- `Mesh file`
- `Training slides`

Problem Domain/Mesh

The geometry we will use for this tutorial is the same as the one used *Basic Heat Conduction*. It consists of three cylinders making a dumbbell shape, with three separate named blocks (ClampA, ClampB, and CenterRod). Blocks (volumetric sections in the mesh) are used in Aria to define different material properties, and can also be used to run post-processors on.



The problem also includes five separate named sidesets (EndA, EndB, EndASides, EndBSides, and RodSurf) applied on different surfaces of the mesh. Sidesets in aria are used to apply boundary conditions or do surface post-processing. Any surface not covered by a sideset will be treated as adiabatic, as will any sideset with no boundary condition applied to it.



The mesh file needed for this tutorial can be downloaded `here`, or it can be generated manually using the Cubit journal file shown in the *Basic Heat Conduction* tutorial.

Convection in Aria

Convection is when a solid exchanges energy with a moving fluid. When the fluid motion is externally driven, this is referred to as *forced convection*. When the fluid motion is driven by buoyancy from heating or cooling it is referred to as *natural convection*. There are a variety of ways to handle convection in Aria, described in broad terms below:

Pre-Computed Convection Coefficient

There are a wide variety of correlations available in the literature for estimating

a convection coefficient for different geometries and fluid properties. When a convection coefficient can be calculated *a priori* one can use a simple flux model where $q = h(T - T_{ref})$. In this case, h and T_{ref} are prescribed by the user as constants or simple functions.

Bulk Node + Correlation
A slightly more advanced option is to use a *bulk node* to solve an energy equation for the fluid along with a convection correlation from the *built-in library*.

Advective Bar + Correlation
This approach solves a 1D fluid flow equation along an *advective bar* which allows variation in temperature in both time and space. Coupled with a convection correlation from the *built-in library* this provides a way to model convection for simple flows.

Full CFD Solve
The most accurate, and most expensive approach is to do a full CFD solve for non-isothermal flow outside the object. This is often done using a coupled Fuego-Aria simulation.

This tutorial will use the pre-computed convection coefficient approach to add convection to the outer surfaces of the dumbbell geometry.

Adding a Convection BC

Scope: Aria Region

We will replace the variable flux we used for `EndB` from the last example with a constant.

```
BC Dirichlet for temperature on EndA = constant value = 300 # K
BC Flux for Energy on EndB = Constant value = -5e4
```

Next, we will create a *mesh group* to make applying the convection boundary conditions easier. The mesh group `clamp_conv` consists of two sidesets, and lets us apply boundary conditions to the mesh group instead of the individual sidesets.

```
Mesh Group clamp_conv = EndASides EndBSides

BC Flux for Energy on clamp_conv = Generalized_Nat_Conv H=5 T_ref=250
BC Flux for Energy on RodSurf = Generalized_Nat_Conv H=10 T_ref=250
```

Since we are applying constant values for h and T_{ref} we can use the *Generalized_Nat_Conv* boundary condition.

Post-process Flux

Scope: Aria Region

To see how much heat is lost due to convection, we will add some new *postprocessors* on the surfaces where we applied the convection boundary condition.

```
Postprocess integrated_flux of equation energy on clamp_conv as␣
 ↪ClampConvLoss
Postprocess integrated_flux of equation energy on RodSurf as RodConvLoss
```

We can also add some post-processed fields to show the heat flux vector ($-k\nabla T$) and the flux field on the `RodSurf` sideset.

```
Postprocess value of expression heat_conduction on all_blocks as HeatFlux
Postprocess value of expression energy_flux on RodSurf as ConvFlux
```

We have now created two new global variables (`ClampConvLoss` and `RodConvLoss`) and two nodal fields (`HeatFlux` and `ConvFlux`). To be able to see these fields, we need to also add them to the results output block.

```
Begin Results Output AriaOutput
    Title Aria: Transient Training Model
    Database Name = heat_cond.e
    At Step 0 Interval = 5
    Nodal Variables = Solution->Temperature as T
    Nodal Variables = ConvFlux
    Nodal Variables = HeatFlux

    Global Variables = FluxA
    Global Variables = FluxB
    Global Variables = ClampConvLoss
    Global Variables = RodConvLoss
End
```

Running Aria

Once you have finished setting up the input file, you are ready to run Aria. From a CEE unix environment, you can load the sierra module to access the latest release of Aria to run it on your local machine. To run it on 4 processors, you could use

```
$ module load sierra
$ launch -n 4 aria -i heat_cond_session2.i
```

Viewing the Results

From a CEE unix environment with graphics, you can launch paraview using

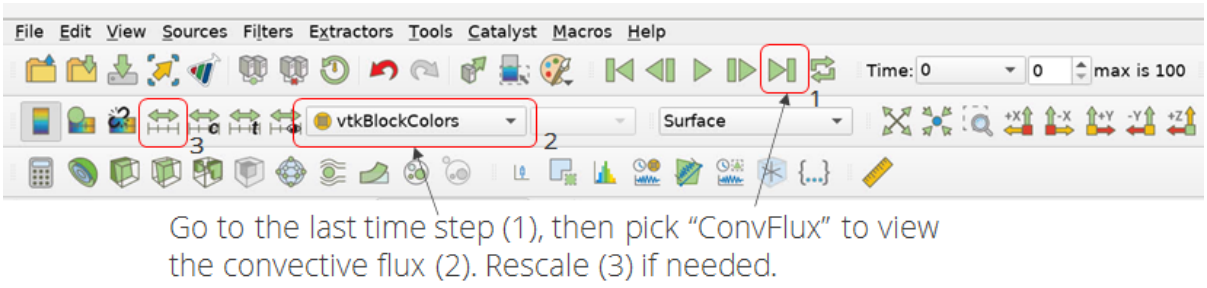
```
$ module load viz
$ paraview heat_cond.e.4.0
```

Or if you have installed Paraview locally, you can launch it and select the exodus file from the appropriate file opening menu.

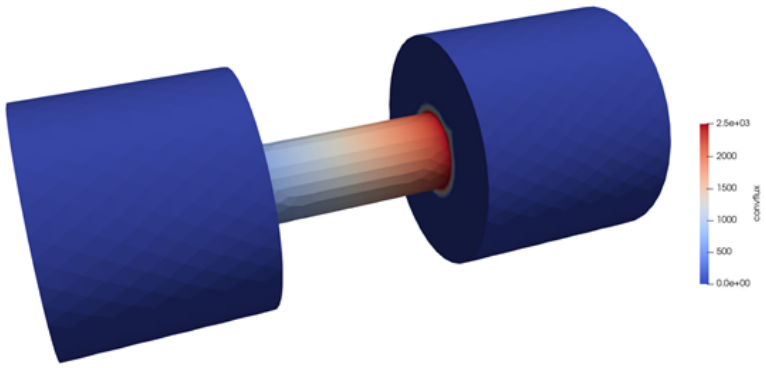
Once you have loaded it, you need to select which variables to show then click `Apply`.

To show the convective flux field (`ConvFlux`) on the center rod, you can

1. Go to the last time step
2. Select the `ConvFlux` variable to show
3. Rescale the color scale if needed

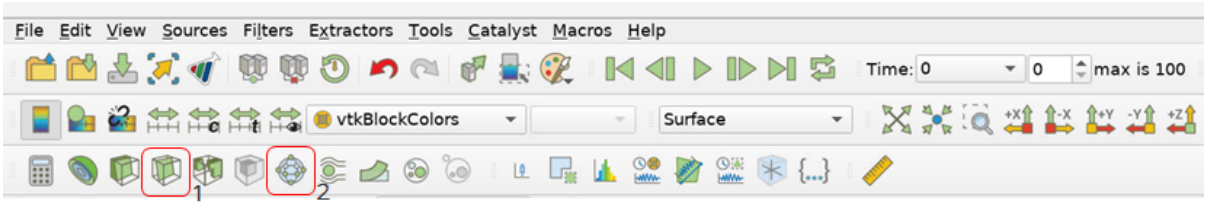


The result shows how the flux varies across the center rod

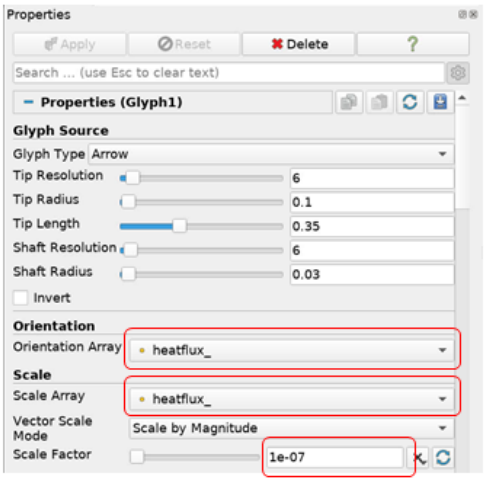


To view the heat flux vector through the center of the domain, you can

1. Slice the domain and click `Apply`
2. Create a `Glyph` filter
3. Choose `HeatFlux` for the orientation and scale array and set a scale factor of 1e-7
4. Click `Apply`

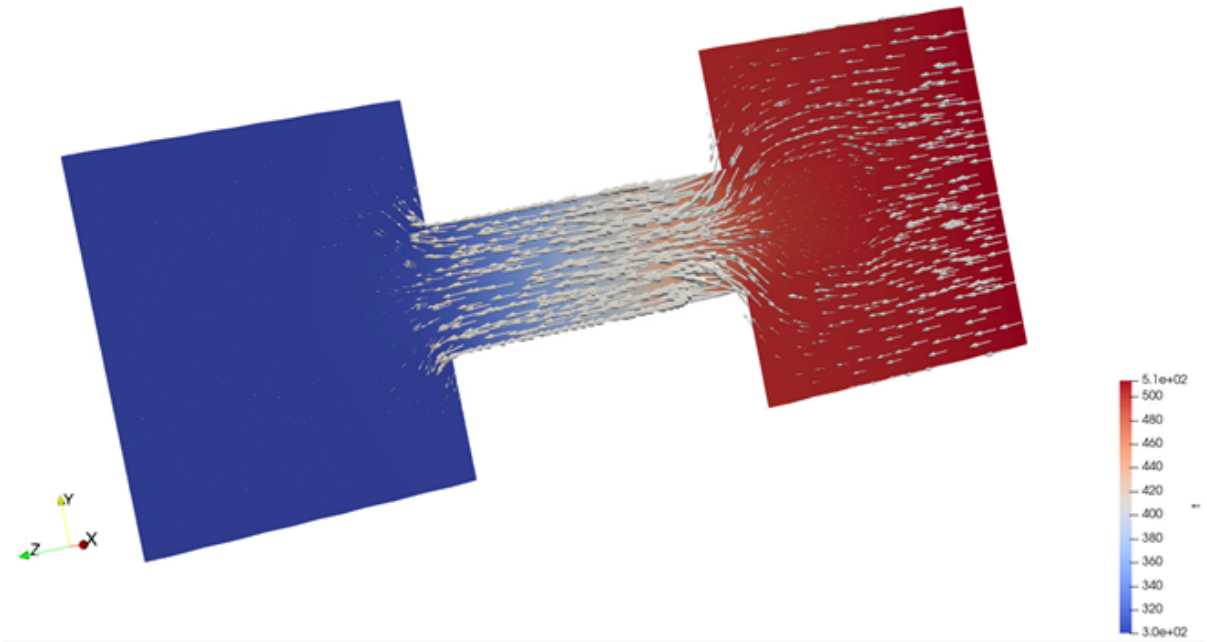


Slice the domain (1) and click "Apply", then apply a Glyph filter (2)



Choose "heatflux_" for the orientation and scale array, and set a scale factor of 1e-7.

Click "Apply".



Radiative Flux BCs

This tutorial builds on the prior *Convective Flux BCs* tutorial and shows how to add a radiative flux boundary condition. This module has also been recorded in prior trainings and is available here or in the player below.

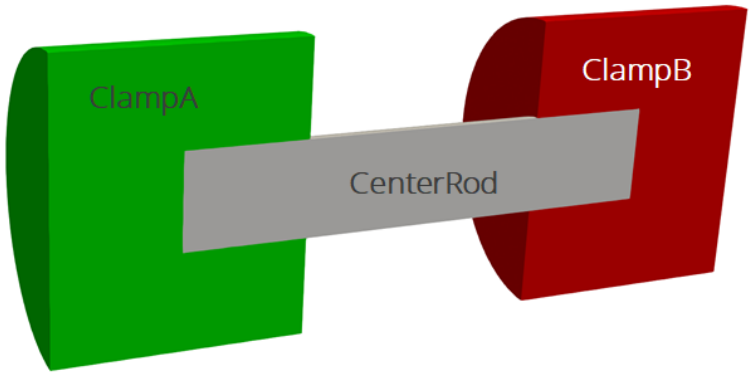
Problem Files

The files required for this tutorial can be downloaded here, or found at `$SIERRA100/TF/Aria` (where `SIERRA100=/projects/sierra100` in a CEE environment, or your Sierra test distribution otherwise).

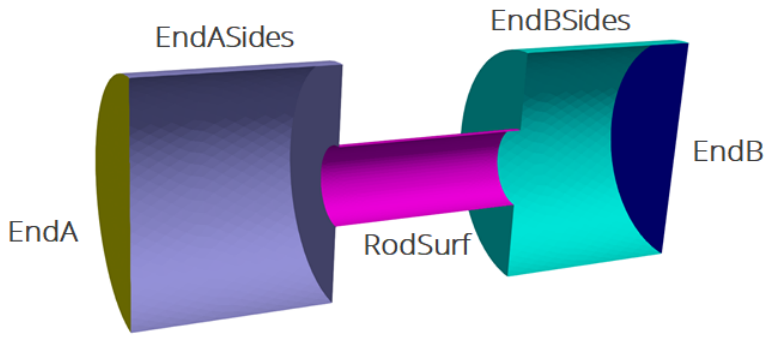
- `Aria input file`
- `Mesh file`
- `Training slides`

Problem Domain/Mesh

The geometry we will use for this tutorial is the same as the one used *Basic Heat Conduction*. It consists of three cylinders making a dumbbell shape, with three separate named blocks (ClampA, ClampB, and CenterRod). Blocks (volumetric sections in the mesh) are used in Aria to define different material properties, and can also be used to run post-processors on.



The problem also includes five separate named sidesets (EndA, EndB, EndASides, EndBSides, and RodSurf) applied on different surfaces of the mesh. Sidesets in aria are used to apply boundary conditions or do surface post-processing. Any surface not covered by a sideset will be treated as adiabatic, as will any sideset with no boundary condition applied to it.



The mesh file needed for this tutorial can be downloaded `here`, or it can be generated manually using the Cubit journal file shown in the *Basic Heat Conduction* tutorial.

Radiation in Aria

There are a few different ways to include thermal radiation into an Aria model, depending on what you are radiating *to* and what you are radiating *through*.

Simple Environment
: For radiative exchange with a simple environment with a known or pre-computed view factor one can apply a simple flux model where $q = \sigma \epsilon F (T^4 - T_{ref}^4)$. In this approach you would prescribe values or models for ϵ, F, and T_{ref}.

Enclosure Radiation
: For radiative exchange between surfaces in your model through an optically thin media you can use the *enclosure radiation* capability. This will calculate view factors between all faces in the enclosure and do a radiosity solve to determine radiative fluxes. The enclosure can be closed or open to a prescribed environment. This option can be expensive for large enclosures since the cost scales with N^2 where N is the number of faces in the enclosure.

Participating Media Radiation
: When the media through which you are radiating is optically thick (absorbs and re-emits instead of being mostly transparent) then you can use a meshed enclosure to *couple Aria and PMR* to do a discrete ordinate solve of the radiative transport equation.

This tutorial will use the simple environment approach to add radiation to the outer surfaces of the dumbbell geometry.

Adding a Radiative BC

Scope: Aria Region

First, we will add a new *mesh group* to make applying the radiative boundary conditions easier. The mesh group `rad_surfs` will include all surfaces except the two on the ends.

```
Mesh Group rad_surfs = all_surfaces - EndA - EndB
BC Flux for Energy on rad_surfs = Generalized_Rad T_ref = 250
```

Since we are applying a constant value for T_{ref} we can use the *Generalized_Rad* boundary condition.

Post-process Flux

Scope: Aria Region

By default, the `ENERGY_FLUX` is the sum of all the applied flux BCs. If we want to *postprocessor* separate contributions from the two flux models we have (convection and radiation) we can pass the model name to the postprocessor.

```
Postprocess value of expression energy_flux on RodSurf Model Generalized_
 ↪Nat_Conv as ConvFlux
Postprocess value of expression energy_flux on RodSurf Model Generalized_
 ↪Rad as RadFlux
```

We will now get a new `RadFlux` nodal field that is just the radiative flux contribution. To be able to see this fields, we need to also add it to the results output block.

```
Begin Results Output AriaOutput
    Title Aria: Transient Training Model
    Database Name = heat_cond.e
    At Step 0 Interval = 5
    Nodal Variables = Solution->Temperature as T
    Nodal Variables = ConvFlux
    Nodal Variables = HeatFlux
    Nodal Variables = RadFlux

    Global Variables = FluxA
    Global Variables = FluxB
    Global Variables = ClampConvLoss
    Global Variables = RodConvLoss
End
```

Running Aria

Once you have finished setting up the input file, you are ready to run Aria. From a CEE unix environment, you can load the sierra module to access the latest release of Aria to run it on your local machine. To run it on 4 processors, you could use

```
$ module load sierra
$ launch -n 4 aria -i heat_cond_session4.i
```

Viewing the Results

From a CEE unix environment with graphics, you can launch paraview using

```
$ module load viz
$ paraview heat_cond.e.4.0
```

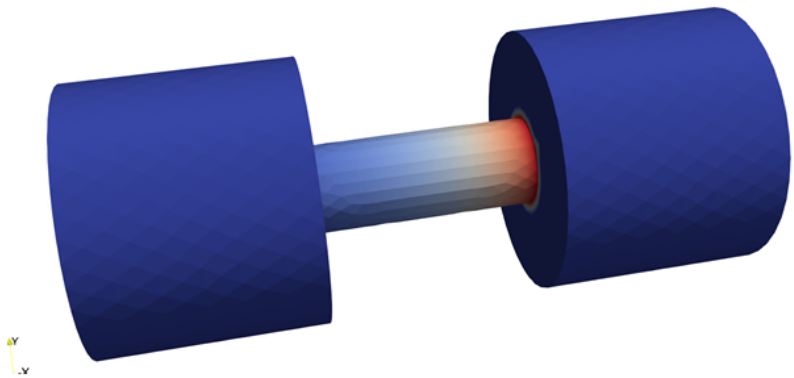
Or if you have installed Paraview locally, you can launch it and select the exodus file from the appropriate file opening menu.

Once you have loaded it, you need to select which variables to show then click `Apply`.

To show the convective flux field (`RadFlux`) on the center rod, you can

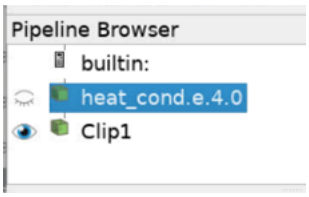
1. Go to the last time step
2. Select the `RadFlux` variable to show
3. Rescale the color scale if needed

The result shows how the radiative flux varies across the center rod



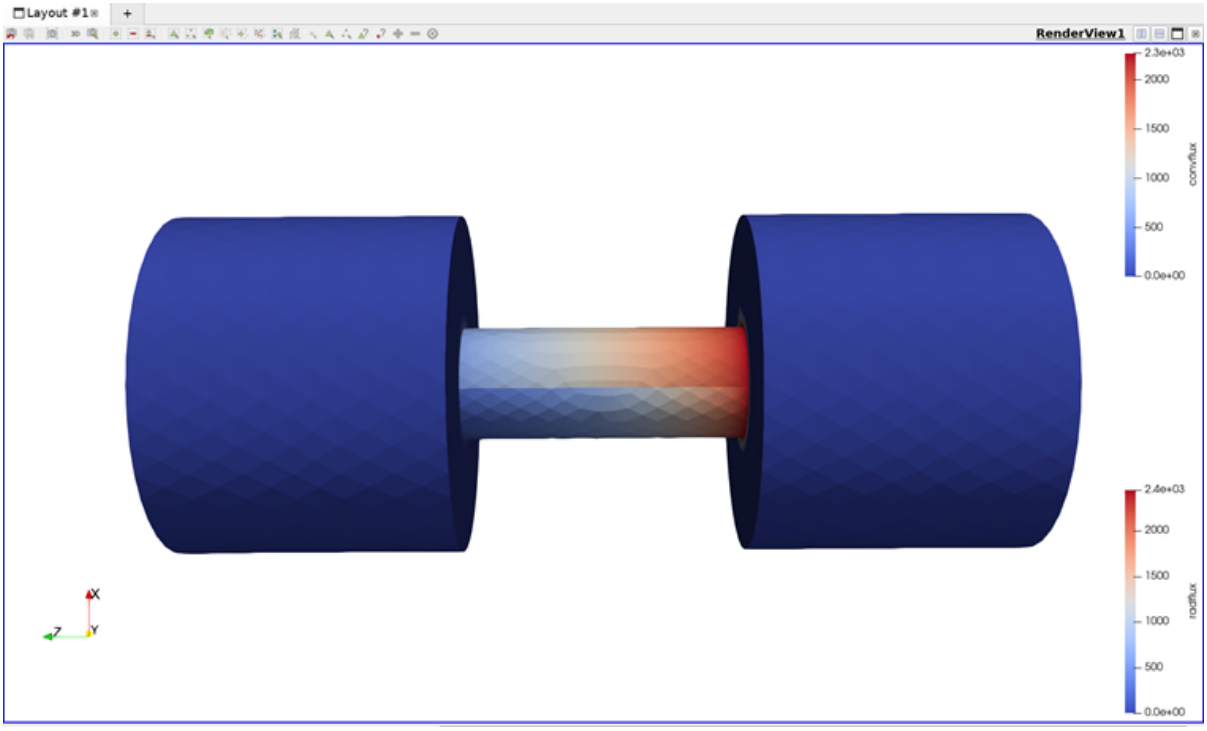
To compare the radiative and convective flux

1. Clip the domain and click `Apply` (with the default clipping normal of `1 0 0`)



2. Re-select the original data set in the pipeline browser
3. Clip the domain again, with the clipping normal set to `-1 0 0` and click `Apply`

4. Choose `RadFlux` to show on the first clipped data set and `ConvFlux` on the second



Multiphysics Coupling

This tutorial builds on the prior *Radiative Flux BCs* tutorial and shows how couple multiple different types of equations (electrical and thermal transport). This module has also been recorded in prior trainings and is available here or in the player below.

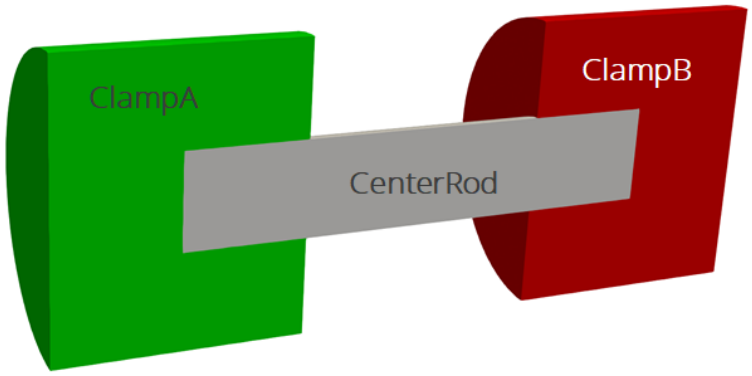
Problem Files

The files required for this tutorial can be downloaded here, or found at `$SIERRA100/TF/Aria` (where `SIERRA100=/projects/sierra100` in a CEE environment, or your Sierra test distribution otherwise).

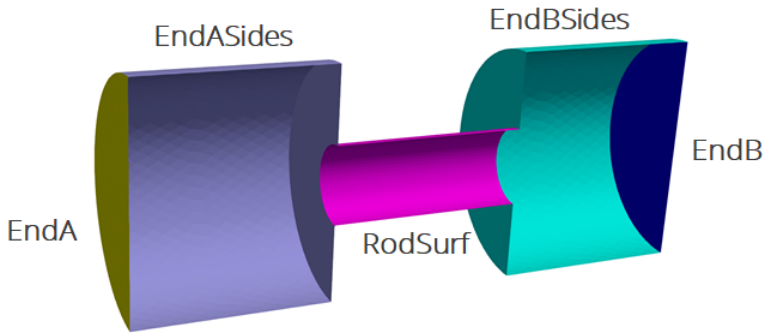
- `Aria input file`
- `Mesh file`
- `Training slides`

Problem Domain/Mesh

The geometry we will use for this tutorial is the same as the one used *Basic Heat Conduction*. It consists of three cylinders making a dumbbell shape, with three separate named blocks (ClampA, ClampB, and CenterRod). Blocks (volumetric sections in the mesh) are used in Aria to define different material properties, and can also be used to run post-processors on.



The problem also includes five separate named sidesets (EndA, EndB, EndASides, EndBSides, and RodSurf) applied on different surfaces of the mesh. Sidesets in aria are used to apply boundary conditions or do surface post-processing. Any surface not covered by a sideset will be treated as adiabatic, as will any sideset with no boundary condition applied to it.



The mesh file needed for this tutorial can be downloaded `here`, or it can be generated manually using the Cubit journal file shown in the *Basic Heat Conduction* tutorial.

Material Updates

Scope: Domain

To add an electrical transport equation, we need to define some additional material properties. These are the `Electrical Conductivity` and `Current Density` (the model for current flux). We must add these properties to both materials in order to solve for electrical transport on both of them.

```
Begin Aria Material aluminum
    Density              = Constant value = 2770  # kg/m^3
    Specific Heat        = Constant value = 800.0 # J/kg-K
    Thermal Conductivity = Constant value = 175.0 # W/mK
    Heat Conduction      = Generalized

    Electrical Conductivity = Constant Value = 10000 # S/m
    Current Density = Ohms_Law
End   Aria Material aluminum
```

```
Begin Aria Material ss304
    Density              = Constant value = 8000
    Specific Heat        = Polynomial Variable=Temperature order=1 C0=500␣
 ↪C1=0.1
    Thermal Conductivity = User_Function name=ssteel_k_function␣
 ↪X=temperature
    Heat Conduction      = Generalized

    Electrical Conductivity = Constant Value = 0.1 # S/m
    Current Density = Ohms_Law
End   Aria Material ss304
```

Solver

Scope: Domain

For the prior examples we used a default thermal solver, but to more robustly solve this multi-physics problem we should use a different preset solver with a more robust preconditioner.

```
BEGIN TPETRA EQUATION SOLVER solve_temperature
  BEGIN PRESET SOLVER
```

(continues on next page)

(continued from previous page)

```
    SOLVER TYPE = Multiphysics
  END
END TPETRA EQUATION SOLVER
```

Governing Equations

Scope: Aria Region

Next we will modify the `EQ` definitions to add a new current equation and add a source term to the energy equation for Joule heating. The source model (`joule_heating`) requires no additional arguments.

```
EQ Energy for Temperature on all_blocks using Q1 with Mass Diff Src
EQ Current for Voltage on all_blocks using Q1 with Diff

Source for Energy on all_blocks = joule_heating
```

This will solve $\nabla\left(-\sigma\nabla V\right) = 0$ for voltage and add an energy source term of $i \cdot \nabla V$ where σ here is the electrical conductivity defined in the materials. These equations are now coupled in one direction since the energy equation depends on the voltage but the voltage solve is independent of temperature. If the electrical conductivity were defined as a function of temperature then it would introduce a two-way coupling between these equations and the current solve would change over time as the temperature changes. No changes to the input would be required to handle this two-way coupling, although the simulation may take more nonlinear iterations to reach convergence.

Boundary Conditions

Scope: Aria Region

The current equation has no `mass` term, so it does not require an initial condition. However, we must supply boundary conditions. For this problem we will set a constant voltage on one end of the domain and a constant current on the opposite end.

```
BC Dirichlet for Voltage on EndA = constant value = 0
BC Flux for Current on EndB = constant value = -100 # Amps/m2
```

Postprocessing

Scope: Aria Region

To see the current flux we can postprocess the `current_density` vector, and to see the energy source due to Joule heating we can postprocess the `energy_source` expression. By default `energy_source` is the sum of all source terms, but individual sources can be postprocessed using the `Model` keyword.

```
Postprocess value of expression energy_source on all_blocks as Esrc
Postprocess value of expression current_density on all_blocks as Current
```

Output

Scope: Aria Region

We now have several new fields to add to the results output block, including the new voltage value (`solution->voltage`) and the postprocessed values.

```
Begin Results Output AriaOutput
    Title Aria: Transient Training Model
    Database Name = heat_cond.e
    At Step 0 Interval = 5
    Nodal Variables = Solution->Temperature as T
    Nodal Variables = Solution->Voltage as V
    Nodal Variables = Current
    Nodal Variables = HeatFlux
    Nodal Variables = ConvFlux
    Nodal Variables = RadFlux
    Nodal Variables = Esrc

    Global Variables = FluxA
    Global Variables = FluxB
    Global Variables = ClampConvLoss
    Global Variables = RodConvLoss
End
```

Running Aria

Once you have finished setting up the input file, you are ready to run Aria. From a CEE unix environment, you can load the sierra module to access the latest release of Aria to run it on your local machine. To run it on 4 processors, you could use

```
$ module load sierra
$ launch -n 4 aria -i heat_cond_session5.i
```

Viewing the Results

From a CEE unix environment with graphics, you can launch paraview using

```
$ module load viz
$ paraview heat_cond.e.4.0
```

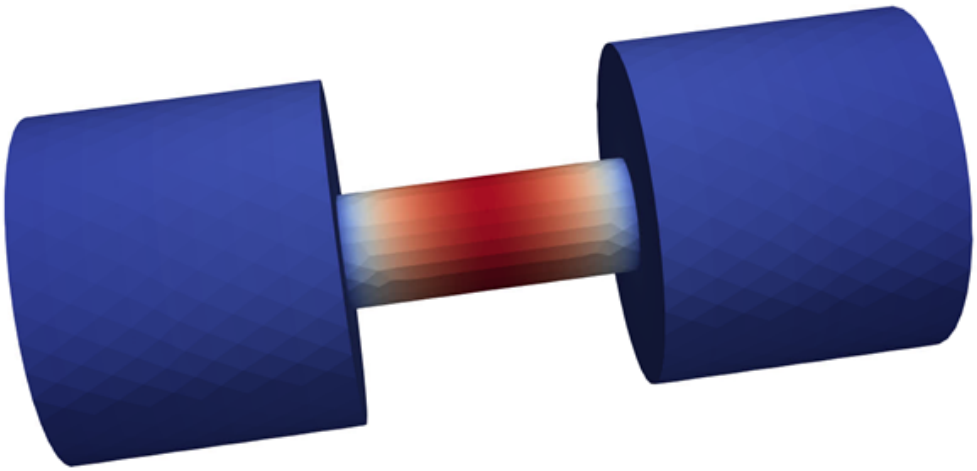
Or if you have installed Paraview locally, you can launch it and select the exodus file from the appropriate file opening menu.

Once you have loaded it, you need to select which variables to show then click “Apply”.

To show the temperature on the center rod, you can

1. Go to the last time step
2. Select the “T” variable to show
3. Rescale the color scale if needed

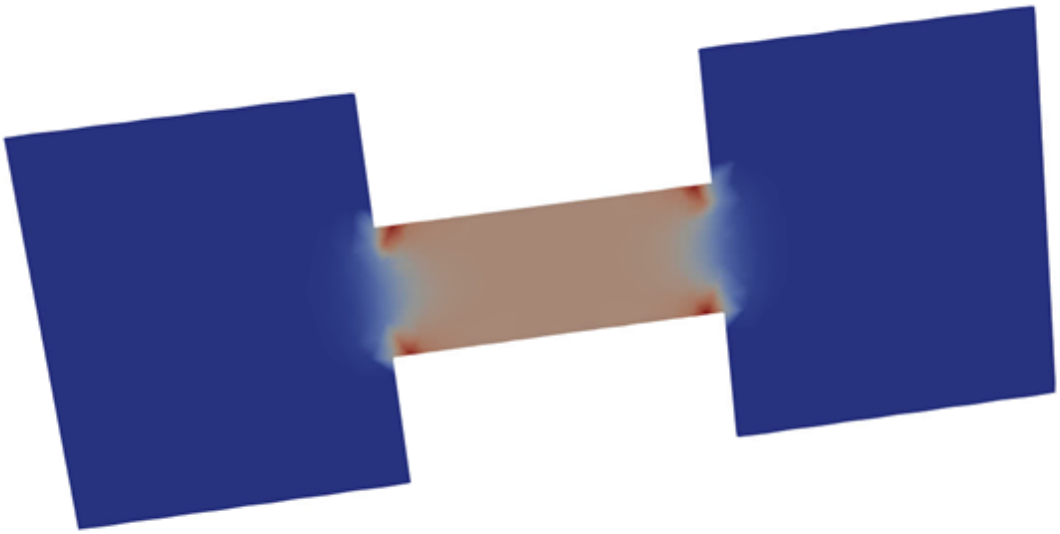
The temperature is highest in the center rod, where the electrical conductivity is lowest.



To view the energy source, you can

1. Slice the domain and click “Apply”
2. Select the “Esrc” variable

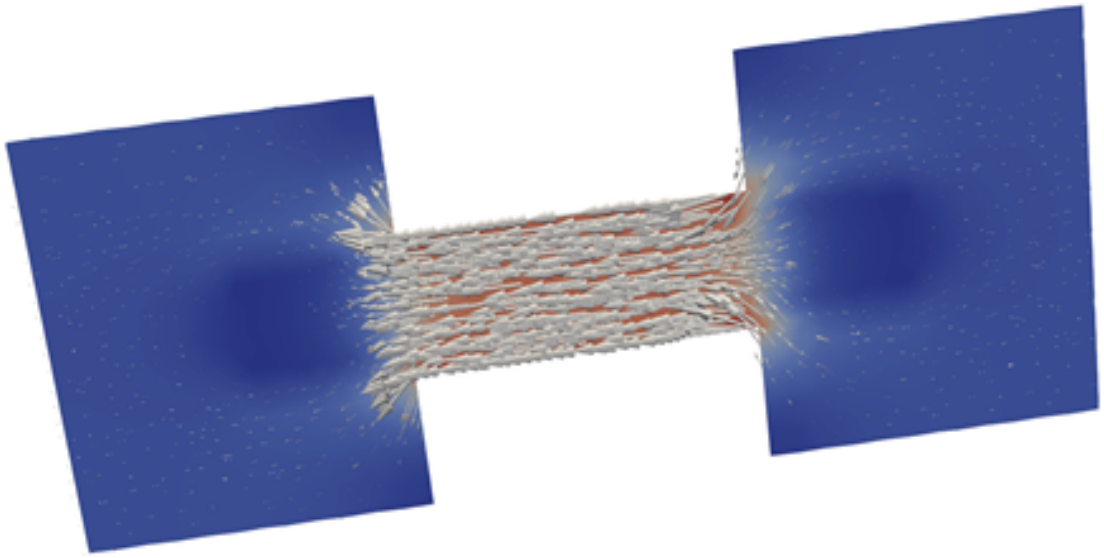
As expected, the energy source is concentrated in the low conductivity center rod.



To visualize the current flux, you can

1. Select the “Glyph” filter
2. Apply a scaling of 1e-5
3. Click “Apply”

The current is highest through the narrow section of the domain, as expected.



Advanced Postprocessing

This tutorial builds on the prior *Multiphysics Coupling* tutorial and shows how to do more advanced postprocessing, extracting values at specific points and evaluating custom functions. This module has also been recorded in prior trainings and is available here or in the player below.

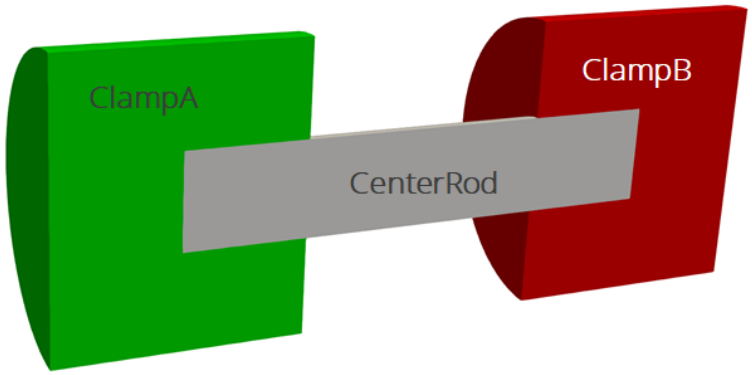
Problem Files

The files required for this tutorial can be downloaded here, or found at `$SIERRA100/TF/Aria` (where `SIERRA100=/projects/sierra100` in a CEE environment, or your Sierra test distribution otherwise).

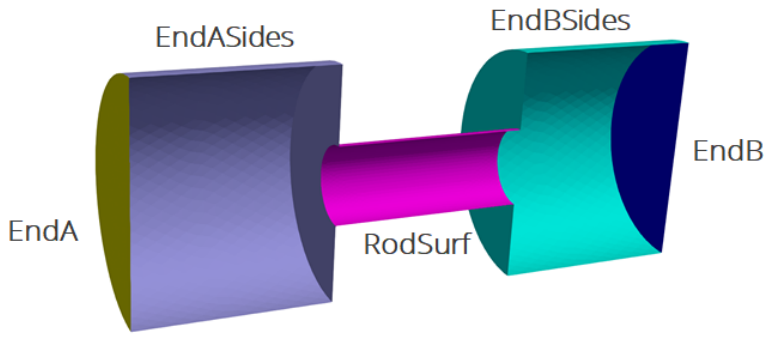
- `Aria input file`
- `Mesh file`
- `Training slides`

Problem Domain/Mesh

The geometry we will use for this tutorial is the same as the one used *Basic Heat Conduction*. It consists of three cylinders making a dumbbell shape, with three separate named blocks (ClampA, ClampB, and CenterRod). Blocks (volumetric sections in the mesh) are used in Aria to define different material properties, and can also be used to run post-processors on.



The problem also includes five separate named sidesets (EndA, EndB, EndASides, EndBSides, and RodSurf) applied on different surfaces of the mesh. Sidesets in aria are used to apply boundary conditions or do surface post-processing. Any surface not covered by a sideset will be treated as adiabatic, as will any sideset with no boundary condition applied to it.

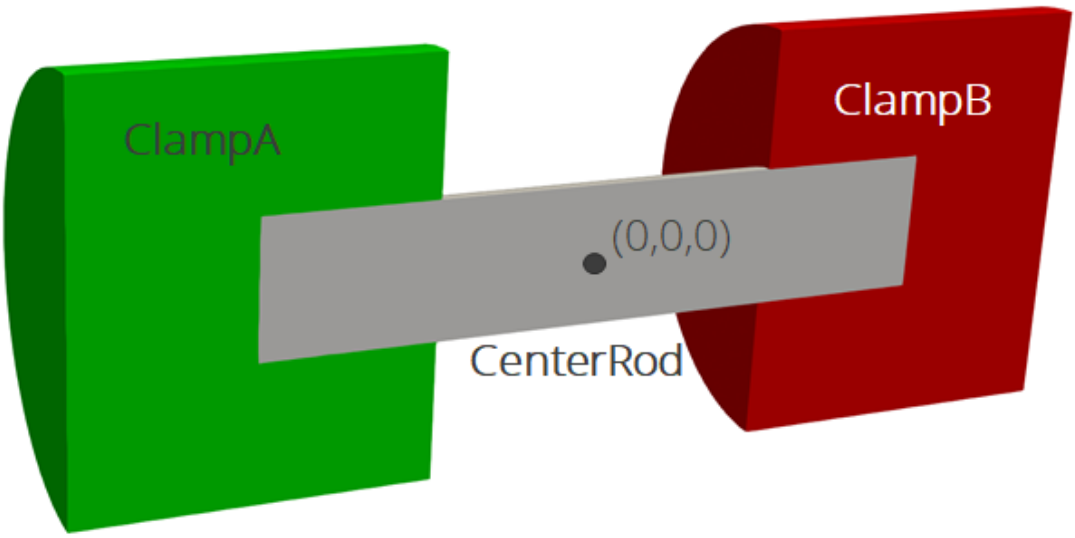


The mesh file needed for this tutorial can be downloaded `here`, or it can be generated manually using the Cubit journal file shown in the *Basic Heat Conduction* tutorial.

Extract Value at a Point

Scope: Aria Region

The first post-processor we will add is to extract a variable at a specific location in the mesh (0,0,0).



To extract the value or an expression (solution DOF or any material property) at a given point, we can use a *data probe*.

```
Postprocess value of expression temperature at point 0 0 0 as Trod
```

Custom Functions

Scope: Aria Region

We can also use *string functions* to post-process any arbitrary quantity we might be interested in for our analysis. For example, if we wanted to calculate the average temperature in Celsius on the center rod and output a field with the ratio of voltage to internal energy, we could define these two post-processors.

```
Postprocess average of function "temperature-273.15" on CenterRod as TavgC
Postprocess value of function "voltage / (density*specific_
↪heat*temperature)" on all_blocks as VEratio
```

Heartbeat Output

Scope: Aria Region

In the prior examples we have output all our data to an Exodus file in the `Results Output` block. However, it is common to output global variables to a text file at a higher frequency.

```
Begin Heartbeat AriaTxtOutput
    Stream name = stats.csv
```

(continues on next page)

(continued from previous page)

```
    Format = csv
    At Step 0 increment = 1
    Variable is Global time
    Variable is Global FluxA
    Variable is Global FluxB
    Variable is Global ClampConvLoss
    Variable is Global RodConvLoss
    Variable is Global Trod
    Variable is Global TavgC
End
```

Results Output

Scope: Aria Region

We created a new nodal field (`VEratio`) with the custom post-processor. To be able to view it, we need to add it to the results output block.

```
Begin Results Output AriaOutput
    Database Name = heat_cond.e
    At Step 0 Interval = 5
    Title Aria: Transient Training Model
    Nodal Variables = Solution->Temperature as T
    Nodal Variables = Solution->Voltage as V
    Nodal Variables = Current
    Nodal Variables = HeatFlux
    Nodal Variables = ConvFlux
    Nodal Variables = RadFlux
    Nodal Variables = Esrc
    Nodal Variables = VEratio

    Global Variables = FluxA
    Global Variables = FluxB
    Global Variables = ClampConvLoss
    Global Variables = RodConvLoss
End
```

Running Aria

Once you have finished setting up the input file, you are ready to run Aria. From a CEE unix environment, you can load the sierra module to access the latest release of Aria to run it on your local machine. To run it on 4 processors, you could use

```
$ module load sierra
$ launch -n 4 aria -i heat_cond_session6.i
```

Viewing the Results

From a CEE unix environment with graphics, you can launch paraview using

```
$ module load viz
$ paraview heat_cond.e.4.0
```

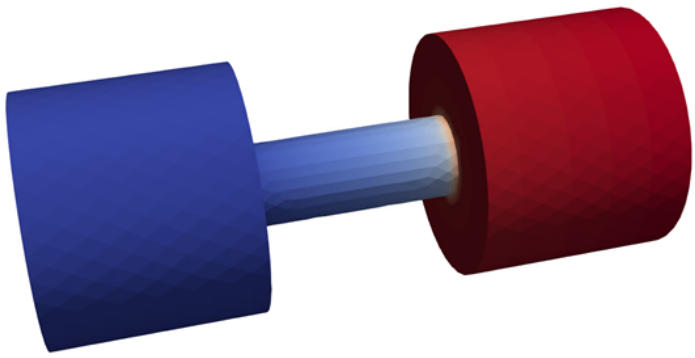
Or if you have installed Paraview locally, you can launch it and select the exodus file from the appropriate file opening menu.

Once you have loaded it, you need to select which variables to show then click "Apply".

To show the post-processed field (`VEratio`), you can

1. Go to the last time step
2. Select the "VEratio" variable to show
3. Rescale the color scale if needed

The result shows the custom calculated quantity.



Note: You can use a calculator filter in Paraview to do custom calculations as well, but you only have access to quantities you already output. In this example, we did not output `specific_heat` or `density`, so we would not be able to calculate `VERatio` in Paraview. It is possible to use Aria to *postprocess additional* quantities based on the already computed results. The next section demonstrates that process to add `specific_heat` and `density` to the results file.

Postprocessing Additional Quantities

As an example of doing additional postprocessing based on existing results we will add `specific_heat` and `density` to a new results file for this case. The input file needed to do this is available from:

- `Aria input file`

The process starts by copying the initial input file and making the following modifications:

1. Add a new finite element model referencing the results of the prior run:

```
Begin Finite Element Model FEModel
    Database Name = mesh.g

    Use Material aluminum for ClampA ClampB
    Use Material ss304 for CenterRod

End   Finite Element Model FEModel

Begin Finite Element Model ExistingResults
    Database Name = heat_cond.e
End   Finite Element Model
```

2. Add an input output region to read the data, transfers to send that data to the Aria region that will do the additional postprocessing, and update the time step settings to let the input output region control the time step. Note that since the meshes are identical, one can use *Include All Blocks* in these transfers rather than have to list each block pairing:

```
Begin Solution Control Description
    Use System Main

    Begin Initialize init
        Advance IORegion
        advance AriaRegion
        transfer existing_to_aria_init
    End
```

(continues on next page)

(continued from previous page)

```
    Begin System Main
        Use Initialize init
        Begin Transient solution_block_1
            Advance IORegion
            Transfer existing_to_aria
            Advance AriaRegion
        End   Transient solution_block_1
    End   System Main

    Begin Parameters for Transient Solution_Block_1
        Start Time       = 0.0        # seconds
        Termination Time = {end_time} # seconds

        Begin Parameters for Aria Region AriaRegion
            Time Step Variation = FIXED
            Initial Time Step Size = {end_time}
            Time Integration Method = BDF2
        End   Parameters for Aria Region AriaRegion

    End   Parameters for Transient Solution_Block_1
End

Begin Input_Output Region IORegion
    Use Finite Element Model ExistingResults
    TIMESTEP ADJUSTMENT INTERVAL = 1
End

Begin Transfer existing_to_aria_init
    Copy Volume Nodes From IORegion to AriaRegion

    Begin Send Blocks
        Include All Blocks
    End
    Begin Receive Blocks
        Include All Blocks
    End

    # Otherwise one can also specify a subset of blocks to transfer
    # Send Block CenterRod to CenterRod
    # Send Block ClampA to ClampA
    # Send Block ClampB to ClampB
```

(continues on next page)

(continued from previous page)

```
    Send field T state none To solution->temperature state new
    Send field V state none To solution->voltage state new
    Send field T state none To solution->temperature state old
    Send field V state none To solution->voltage state old
End

Begin Transfer existing_to_aria
    Copy Volume Nodes From IORegion to AriaRegion

    Begin Send Blocks
        Include All Blocks
    End
    Begin Receive Blocks
        Include All Blocks
    End

    Send field T state none To solution->temperature state new
    Send field V state none To solution->voltage state new
End
```

3. Switch the Aria equation definitions to XFER and disable the boundary conditions:

```
EQ Energy for Temperature on all_blocks using Q1 with XFER #Mass Diff Src
EQ Current for Voltage on all_blocks using Q1 with XFER #Diff

IC for Temperature on all_blocks = constant value = 300 # K

#Source for Energy on all_blocks = joule_heating

#BC Dirichlet for temperature on EndA = constant value = 300 # K
#BC Dirichlet for temperature on EndB = constant value = 300 # K

Mesh Group clamp_conv = EndASides EndBSides
Mesh Group rad_surfs = all_surfaces - EndA - EndB

#BC Flux for Energy on clamp_conv = Nat_Conv H=5 T_ref=250
#BC Flux for Energy on RodSurf = Nat_Conv H=10 T_ref=250

#BC Flux for Energy on rad_surfs = Rad T_ref = 250 Crad = {e*sigma}

#BC Dirichlet for Voltage on EndA = constant value = 0
#BC Flux for Current on EndB = constant value = -100 # Amps/m2
```

4. Add the additional desired postprocessors and include them in the results output:

```
Postprocess value of expression specific_heat on all_blocks as Cp
Postprocess value of expression density on all_blocks as rho

...

Begin Results Output AriaOutput

    ...

    Nodal Variables = Cp
    Nodal Variables = rho

    ...
End
```

5.2 Examples

Similarly, the following list contains a set of common example problems to use as a starting point for your analyses

Thermal Examples

Advective Bar Example

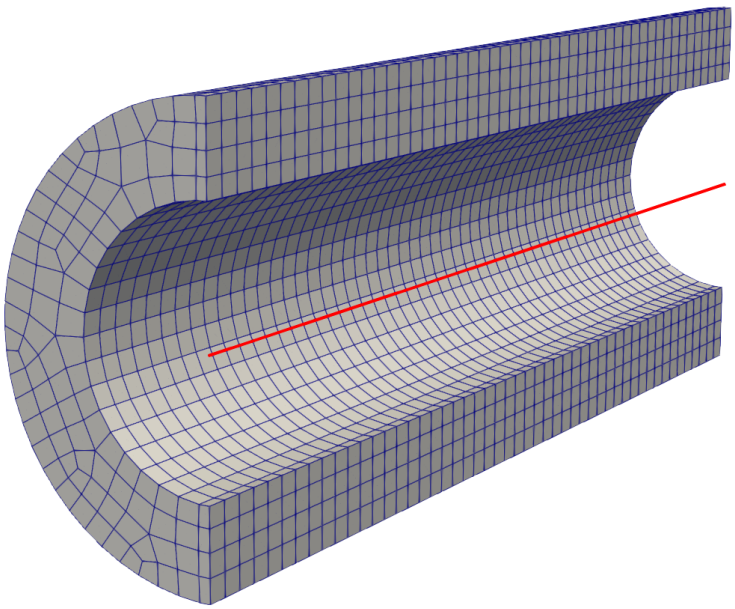
An advective bar is a 1D fluid solve that accounts for changes in fluid temperature along the bar and can be used to provide a spatially varying convective environment.

Problem Files

The files required for this example can be downloaded here, or found at `$TESTS/aria_rtest/example/training/advectivebar` (where `TESTS=/projects/sierra/tests` in a CEE environment, or your Sierra test distribution otherwise).

- `Input File`
- `Mesh journal file`

This problem uses a simple pipe geometry with a bar down the center of the pipe.



Input Deck Modification

To add an advective bar to an Aria problem, you need to make the following changes:

1. Add a block of 1D bar elements to the mesh file with a nodeset on the "entrance node". For example, the commands to add a bar in Cubit are:

```
# Create the bar (a line)
create curve location 0 0 -0.25 direction 0 0 1 length 0.5

# Mesh the bar
curve 7 interval 100
curve 7 scheme equal
mesh curve 7

# Create a nodeset on the entrance vertex
nodeset 1000 add vertex in curve 7 with z_coord > 0.0
nodeset 1000 name "bar_inlet"

# Create a block for the bar elements
block 2 add curve 7
block 2 element type bar2
block 2 name "bar"
```

2. Define functions for the bar geometry and bar fluid density

```
begin definition for function bar_area
    scale by 1.0
    type = piecewise linear
    begin values
      -1.   {PI/4*D^2}
       1.   {PI/4*D^2}
    end values
end

begin definition for function bar_hydraulic_diameter
    scale by 1.0
    type = piecewise linear
    begin values
      -1.   {D}
       1.   {D}
    end values
end

begin definition for function air_density
    type = piecewise linear
    begin values
{NOECHO} { _T = 200}
{loop(12)}
{NOECHO}      { _T = _T + 25}
{ECHO}        {_T}  {101325*28/(8314*_T)}
{endloop}
    end values
end definition for function air_density
```

3. Define a material to use for the fluid in the bar

```
BEGIN Aria MATERIAL Air
  Viscosity            = constant value = 1e-5
  density              = user_function name = air_density   X =␣
↪temperature
  specific heat        = constant value = 1000
  thermal conductivity = constant value = 0.04
  bar area             = user_function name = bar_area X = bar_
↪coordinate
  heat conduction = basic
  Advection Velocity = bar
END Aria MATERIAL Air
```

And assign it to the advective bar block in the finite element model section

```
begin finite element model fem
  # ...
  use material Air for bar
end finite element model fem
```

4. Add an advective bar block in the Aria region and define an energy equation on it

```
Begin Advective Bar Network airBar
   add volume bar
   initial temperature = 300
   MASS FLOW RATE AT bar_inlet = 0.1 # kg/s

   coordinate reference = bar_inlet component z
   fluid density function = air_density
   hydraulic diameter function = bar_hydraulic_diameter
   flow cross sectional area function = bar_area
end

EQ ENERGY for TEMPERATURE on bar using Q1P0 with Diff Lumped_
 ↪mass Adv SUPG
BC Dirichlet for temperature on bar_inlet = constant value = 500
IC for temperature on bar = constant value = 500
```

5. Add a convective boundary condition coupling the solid domain with the bar

```
BC Bar_Flux for energy on inner_surface = generalized_nat_conv␣
 ↪bar=airBar h=500.0
```

As with the bulk node - the convection coefficient can be defined using an inline string function or using a correlation (specified with the `hcorr` argument).

Bulk Node Example

A bulk node provides a method for tracking the temperature in a non-meshed region in the problem. This could be an external environment, or an internal cavity filled with a fluid of some kind. A bulk node adds a single DOF to the overall equation system and the surfaces tied to the bulk node can exchange energy convectively.

Because bulk nodes can use the *convective correlation library*, using a bulk node with an extremely large volume as the ambient condition allows the use of the correlation library for not only internal surfaces but also those convecting to an external ambient environment.

Problem Files

The files required for this example can be downloaded here, or found at `$TESTS/aria_rtest/example/training/bulkelem` (where `TESTS=/projects/sierra/tests` in a CEE environment, or your Sierra test distribution otherwise).

- `Input File`
- `Mesh`

Input Deck Modification

To add a bulk node to an Aria problem, you need to make the following changes:

1. Add a material to use in the bulk node. At a minimum, this material should define the necessary properties to solve an energy equation on the bulk node (density and specific heat). If using a correlation for the convection coefficient, additional properties may be required.

```
Begin Aria Material air
    Density       = Constant value = 0.972  # kg/m^3
    Specific Heat = Constant value = 1004.2 # J/kg-K
End
```

2. Add a `Bulk Fluid Element` block in the Aria region and specify the material to use on it, its initial temperature, the equation to solve, and how to calculate the bulk node volume. The volume can either be computed automatically from the bounding surfaces of the bulk node (defined based on which boundary conditions are linked to this specific bulk node) or specified manually using `Bulk Element Volume = Constant v = 1.23` in the `Bulk Fluid Element` block.

```
# Aria Region
Begin Bulk Fluid Element air_void
    Calculate Volume from Enclosing Surface
    Material = air
    Initial Temperature = 300.0 # K
    Bulk EQ Energy for Temperature Using P0 with Mass
End   Bulk Fluid Element air_void
```

3. Link convection boundary conditions to the bulk node. If the bulk node volume is calculated from its enclosing surfaces, those are determined based on which boundary conditions are linked to the bulk node. Boundary conditions can be changed from *flux* type to *bulknode_flux* type. The convection coefficient can either be supplied as an in-line string function, using the `heat transfer`

coefficient property in a *surface material*, or using a *correlation*. When using a correlation, specify which correlation block to use using the hcorr argument.

```
BC Bulknode_Flux for Energy on bulk_node_surfs = Generalized_Nat_
↪Conv \$
    bulk_node = air_void \$
    h = 10.0
```

```
BC Bulknode_Flux for Energy on bulk_node_surfs = Generalized_Nat_
↪Conv \$
    bulk_node = air_void \$
    hcorr = my_convection_correlation
```

The boundary condition can also be set using a block-style syntax

```
Begin Convective Flux Boundary Condition conv_to_void
    Add Surface bulk_node_surfs
    Convective Coefficient = 10.0
    Use Bulk Element air_void
End
```

CHEMEQ Example

If your problem has a chemically reacting material, you can define a *CHEMEQ* model in the material definition to track species evolution and calculate a heat source and gas production from the material decomposition.

When using CHEMEQ, Aria automatically creates multi-valued element fields to track species concentration at each integration point (or, optionally, at the element centroid). These species cannot be transported, but are integrated in time using the reaction mechanism and ODE solver you define and can be used to define other derived quantities (heat source, composition-dependent material properties, gas production for pressurization).

Problem Files

The files required for this example can be downloaded here, or found at $TESTS/aria_rtest/example/training/chemistry (where TESTS=/projects/sierra/tests in a CEE environment, or your Sierra test distribution otherwise).

- Input File
- Mesh

Input Deck Modification

There are several steps required to add CHEMEQ to your problem.

1. First, add a CHEMEQ block inside the relevant material to define the chemistry mechanism. Refer to the *CHEMEQ* section in the manual for details about how to define this block.

```
Begin Aria Material foam
    # ...
    Begin Parameters for CHEMEQ model foam_reaction_model

        Species Names    = F_SPECA   F_SPECB
        Species Phases   = CONDENSED  GAS

        Concentration Units = mass_fraction
        Energy release units are per unit mass

        Begin Reaction R1
          Reaction is F_SPECA -> F_SPECB
          Rate Function = Arrhenius A = 1e14 Ea = 188370 R = 8.
↪314
          Concentration Function = Standard mu = Automatic
          Heat of Reaction = -4186800 # J/kg
        End

    End   Parameters for CHEMEQ model foam_reaction_model
End   Aria Material foam
```

2. (Optional) Create expressions from the CHEMEQ state variables in order to define composition-dependent properties or allow post-processing of CHEMEQ variables using Aria *expression postprocessors*.

```
Begin Aria Material foam
    # ...
    Species Names = F_SpecA F_SpecB
    Mass Fraction of F_SpecA = From_Chemeq
    Mass Fraction of F_SpecB = From_Chemeq

    Thermal Conductivity = Scalar_String_Function f = "0.5 - 0.
↪1*mass_fraction_f_specB"

    Begin Parameters for CHEMEQ model foam_reaction_model
        # ...
    End
```

(continues on next page)

(continued from previous page)

```
End
```

3. Define an energy source from the CHEMEQ model. The name of the CHEMEQ model is not required - Aria can determine this automatically based on the material assigned to the block.

```
Source for Energy on reacting_foam = ChemEQ_Heating
```

Note: Adding an energy source is required in order to activate the CHEMEQ model - a CHEMEQ model with no corresponding energy source term will not be integrated.

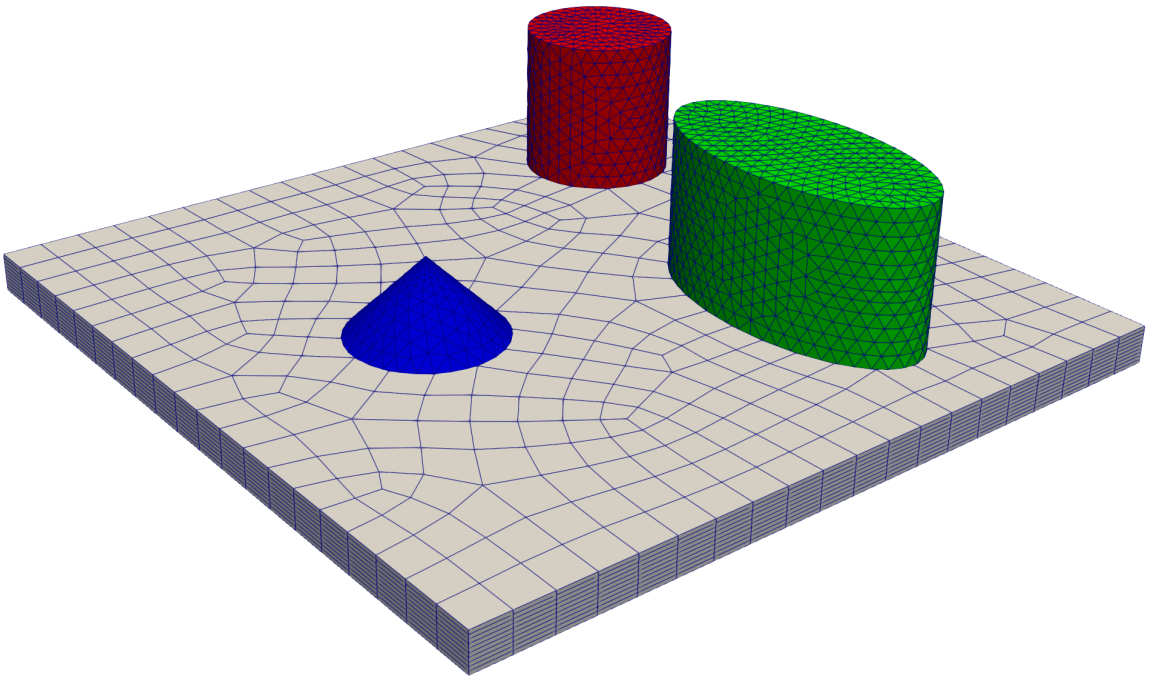
4. Define a CHEMEQ solver block with the *ODE solver* specification and initial conditions for species concentrations. The name used for this block must match the name given to the CHEMEQ block (`foam_reaction_model` here). This block can also be used to define activation and deactivation conditions for the CHEMEQ mechanism to turn it on or off when certain temperature conditions are met.

```
Begin CHEMEQ Solver for foam_reaction_model

    ODE Solver = CVODE ADAMS 12 FUNCTIONAL
    Absolute Tolerance = 1e-10
    Relative Tolerance = 1e-4
    Maximum Substeps = 100000

    # Set the initial concentration values
    Species F_SPECA = 1.0
    Species F_SPECB = 0.0
End
```

Contact Example

Aria supports using *contact* to tie the temperature across adjacent interfaces that are not meshed contiguously. For complicated geometries, it is often easier to create quality meshes when the meshes are allowed to be discontiguous between parts.

For example, in the scenario shown below it is desirable to use thin hex elements in the board to accurately capture flux through the board, but using hexes in the shapes mounted on the board may be less feasible. Thermal contact can be used to allow non-contiguous meshes at those interfaces and make the problem setup easier.



When making such a mesh, the different volumes can be imprinted (but not merged) to force the separate blocks to use contiguous edges. In this case, this forces the mesh on the plate to have edges conformal with the component edges so faces are either fully in contact or fully exposed - although this is not required in order to use contact.

Problem Files

The files required for this example can be downloaded here, or found at `$TESTS/aria_rtest/example/training/contact` (where `TESTS=/projects/sierra/tests` in a CEE environment, or your Sierra test distribution otherwise).

- `Input File`
- `Mesh`

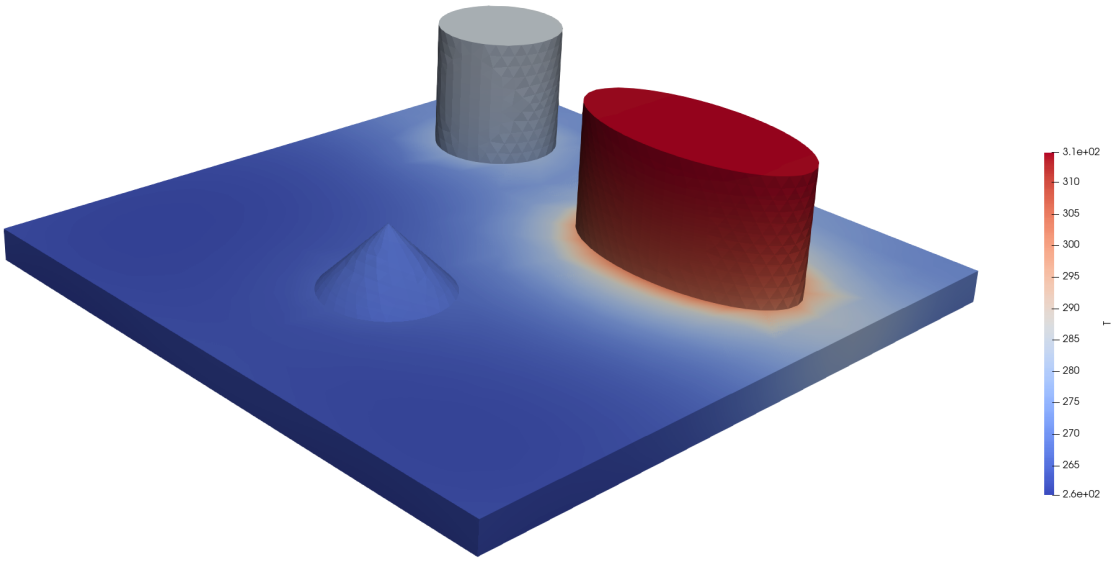
Input Deck Modification

Regular boundary conditions (like convection) are automatically excluded from faces that are in contact, so nothing special needs to be done for boundary conditions on surfaces that may be in partial contact. In this example, `sideset_2` is the entire top surface of the board, including the portions in contact with the components, but we can safely apply a convection BC to that sideset and it will only be applied to the areas that are not in contact. If this is not the desired behavior, one can use a `no_coverage_flux` BC type to skip this masking and apply the boundary condition everywhere regardless of contact status.

In this case, using block skinning is the easiest approach, and the only change to the input file is the addition of the following block in the Aria region.

```
Begin Contact Definition res1
    Begin Search Options
        normal tolerance = 0.001
    End

    # All skinned blocks participate
    skin all blocks = on

    begin enforcement enf_1
        Enforcement for Energy = tied_temperature
    end
End
```

The result is that energy can pass between the different components across the contact interfaces.



Directed Energy (Laser) Example

The *directed energy* source block provides a convenient way of defining a source or flux from a moving laser. This example shows how to add a Gaussian weld flux that moves across a domain.

Problem Files

The files required for this example can be downloaded here, or found at `$TESTS/aria_rtest/example/training/directedEnergy` (where `TESTS=/projects/sierra/tests` in a CEE environment, or your Sierra test distribution otherwise).

- `Input File`
- `Mesh`

Input Deck Modification

The only things needed to add a laser flux are a path function for the movement of the laser and a `BEGIN DIRECTED ENERGY` block in the Aria region or thermal equation system.

1. Define a multi-column function for the laser path (at the Sierra level). The columns define the laser spot location and orientation over time.

```
# Sierra level
Begin Definition for Function PATH3
  Type is multicolumn piecewise linear
  Column Titles Time X Y Z NX NY NZ
  Begin Values
    75.0   0.16 -0.4333    -0.52   0.      0.    -1.0
    150.0  1.5  -0.1624875 -0.52   0.      0.    -1.0
  End
End
```

Note: Unlike user functions used for material properties, laser path functions do not clip time to the function domain. Instead, the laser is off outside the time bounds of the function.

2. Add a directed energy block in the Aria region or thermal equation system

```
# Aria Region level
BEGIN DIRECTED ENERGY MODEL bc3
  laser_power = CONSTANT Value=1082.0

  add surface surface_2
  Origin and Normal function = path3

  BEGIN Gaussian Power Weld Flux
```

(continues on next page)

(continued from previous page)

```
    efficiency = 1.0
    beam diameter = 0.20
    effective beam diameter = 0.18
  END
END
```

This will produce a flux on *surface_2* based on the power and path information provided in the block above. It can be *post-processed* with the following command

```
Postprocess value of expression energy_flux on surface_2 as my_flux
```

or, if you have multiple fluxes on that surface and want only the laser contribution

```
Postprocess value of expression energy_flux on surface_2 model gaussian_
↪power_weld as my_flux
```

The same type of post-processor can be applied to laser sources, as

```
Postprocess value of expression energy_source on block_1 as my_source
```

with the same ability to restrict the post-processor to the specific model when needed.

Enclosure Example

If your problem has non-meshed internal voids and you want to model radiative heat transfer through these voids, you need to define an *enclosure*.

Problem Files

The files required for this example can be downloaded here, or found at `$TESTS/aria_rtest/example/training/enclrad` (where `TESTS=/projects/sierra/tests` in a CEE environment, or your Sierra test distribution otherwise).

- `Input File`
- `Mesh`

Input Deck Modification

There are a few things that need to be modified in an input deck to include enclosure radiation. These are

1. Ensure that the `Global Constants` block is present at the sierra level and includes the Stefan Boltzmann constant

```
# Sierra level
Begin Global Constants
  Stefan Boltzmann Constant = 5.67e-08 # W/m2-K4
End   Global Constants
```

2. Add a viewfactor calculation block to define which algorithm to use for calculating view factors.

```
# Aria Region level
Begin Viewfactor Calculation vf_hemicube
  Compute Rule = HEMICUBE
End
```

3. Add a viewfactor smoothing block to define which method (if any) to use for view factor smoothing.

```
# Aria Region level
Begin Viewfactor Smoothing vf_smooth
  Method = LEAST-SQUARES
End
```

4. Add a radiosity solver block to define what linear solver settings to use when solving the enclosure radiosity equation.

```
# Aria Region level
Begin Radiosity Solver rad_solver
  Coupling = MASON
  Solver = CHAPARRAL CG
  Convergence Tolerance = 1e-06
  Maximum Iterations = 200
End
```

5. Add the actual enclosure definition block, which defines the surfaces that bound the enclosure, which viewfactor calculation and smoothing models to use (from steps 2 and 3), and which radiosity solver to use (from step 4).

```
# Aria Region level
Begin Enclosure Definition cavity_1
    Add Surface surface_401 surface_402
    Add Surface surface_403 surface_404

    Use Viewfactor Calculation vf_hemicube
    Use Viewfactor Smoothing vf_smooth
    Use Radiosity Solver rad_solver
End
```

6. (optional) Ensure that the materials adjacent to an enclosure have a definition for emissivity. Alternately, this can be supplied in the enclosure definition block, but using the material properties is recommended (if you change a block from one material to another, the enclosure will use the correct emissivity for the new material).

```
Begin Aria Material aluminum
  Emissivity  = Constant value = 0.2
  # ...
End
```

If you have multiple enclosures and want to use the same settings for all of them, you do not have to repeat steps 2 to 4, just add additional enclosure definition blocks that refer to the same viewfactor calculation, smoothing method, and radiosity solver blocks.

Partial Enclosure Example

If your problem has non-meshed internal *enclosures* that are not fully closed, you can use a *partial enclosure* to allow the open parts of the enclosure to radiate to a specified far-field condition based on the view factor of the opening.

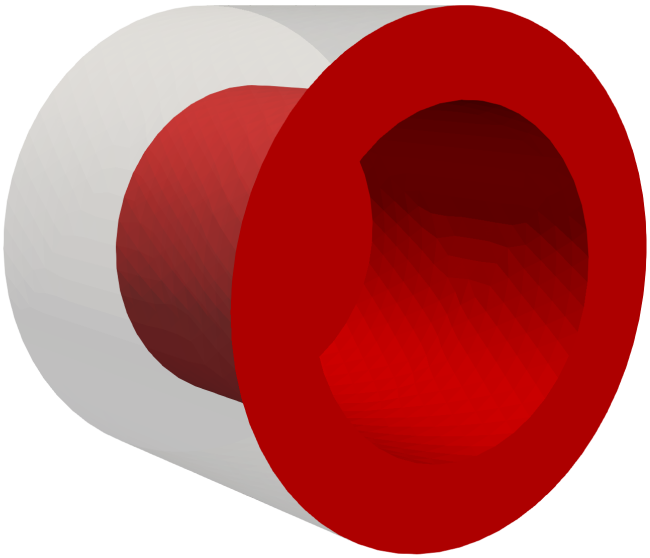
Warning: Gaps in a partial enclosure (e.g. near contact surfaces or from forgetting to add a sideset to the enclosure definition) will be treated as part of the enclosure opening and allow radiation to the far-field condition through those openings. Ensure you have well-formed enclosures when using the partial enclosure feature.

Problem Files

The files required for this example can be downloaded here, or found at `$TESTS/aria_rtest/example/training/enclrad` (where `TESTS=/projects/sierra/tests` in a CEE environment, or your Sierra test distribution otherwise).

- `Input File`
- `Mesh File`

The geometry for this example is a cylinder with a cup open on one end. The cup and ends of the cylinder (red surfaces) make a partial enclosure that is open to a specified environment. Faces inside the cup can radiate to other parts of the cup, and to the environment.



Input Deck Modification

To use a partial enclosure, you must make the following changes to the existing enclosure definitions.

1. Define a viewfactor smoothing block that uses no smoothing (smoothing partial enclosures is not allowed)

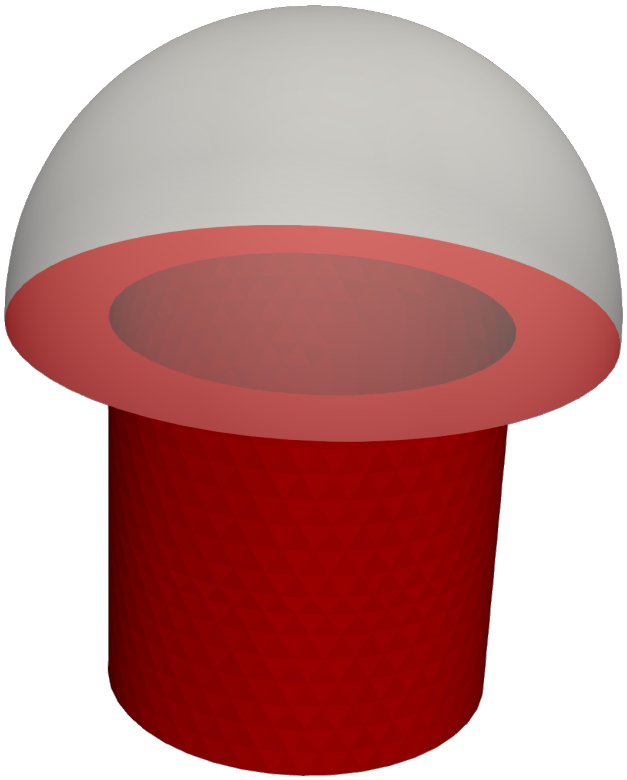
```
# Aria Region level
Begin Viewfactor Smoothing vf_smooth_none
  Method = NONE
End
```

2. Define the partial enclosure area, temperature, and emissivity in the enclosure definition block - make sure to set the smoothing to use the block you defined with no smoothing.

Note: The partial enclosure area is always required, but has no effect on the solution when the partial enclosure emissivity is 0 or 1.

```
# Aria Region
Begin Enclosure Definition enc1
  Add Surface surface_1

  Use Viewfactor Calculation vf_hemicube
  Use Viewfactor Smoothing vf_smooth_none
  Use Radiosity Solver rad_solver

  # Define partial enclosure
  partial enclosure emissivity = 1.0
  partial enclosure area = 1 # arbitrary since emissivity = 1
  partial enclosure temperature = 50.0
End
```

Warning: When using the partial enclosure to represent a real object with `0 < emissivity < 1` you must use the object's real area or the results will be incorrect.
For example, if the partial enclosure environment is really a hemisphere surface of aluminum using real aluminum emissivity (as in the image below) but you want to use a partial enclosure since it will be held at a constant temperature, you must still set the area to the actual area of the hemisphere you want to model.



OMD with Porous Flows Example

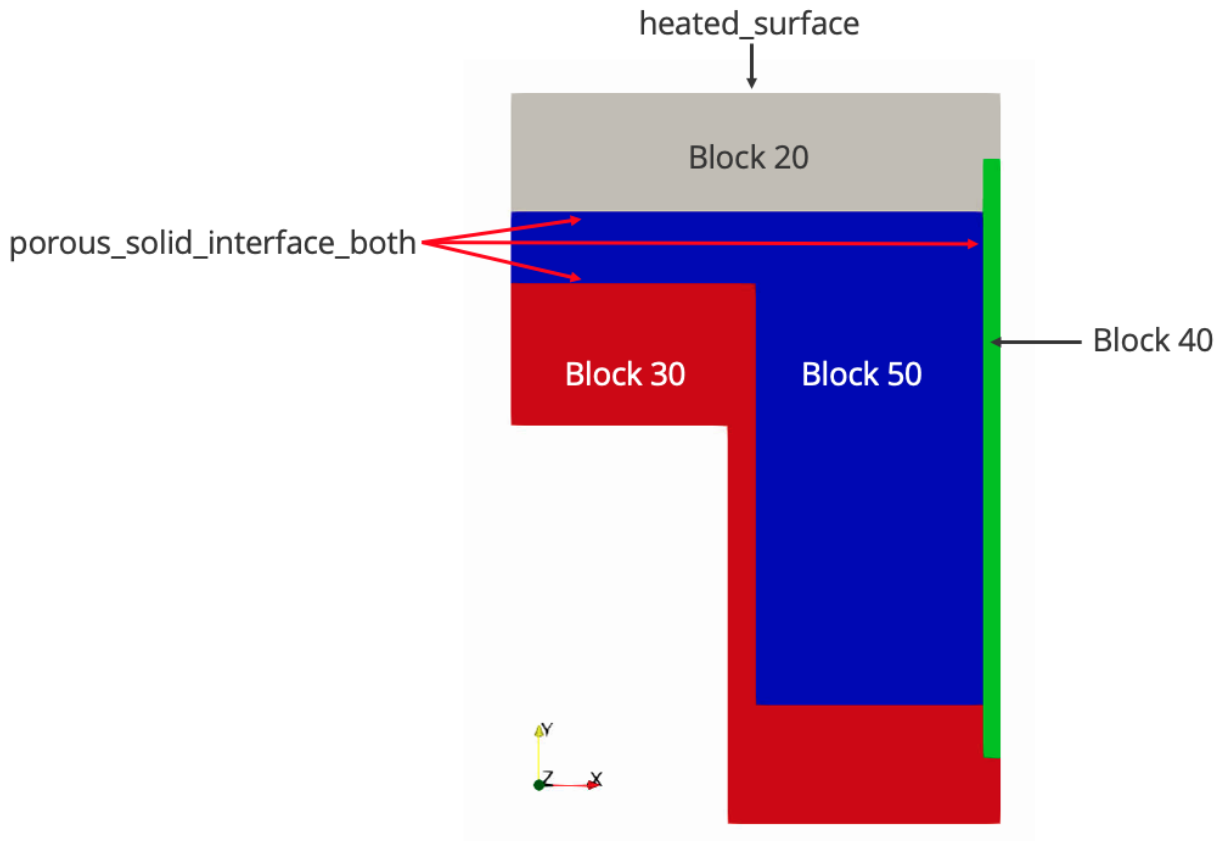
This example shows how to set up an OMD example problem with porous flows; this particular example will be in 2D and use an *axisymmetric coordinate system*.

Problem Files

The files required for this example can be downloaded here, or found at `$TESTS/aria_rtest/example/training/omd_porous_flow_fic` (where `TESTS=/projects/sierra/tests` in a CEE environment, or your Sierra test distribution otherwise).

- `Input File`
- `Mesh`

The problem uses a 2D mesh that is axisymmetric about the y-coordinate. An example of the geometry mesh with annotations of block/sidesets is shown below:



Block 20, 30, and 40 are a stainless steel material, and block 50 is a foam material. A time dependent dirichlet boundary condition is defined on the `heated_surface` mesh group to induce the chemistry kinetics. A `porous_solid_interface_both` mesh group, which is the boundary between the foam (block 50) and and stainless steel (blocks 20, 30 and 40), is created to define a contact flux.

Input Deck Modification

There are several steps required to set up this model. One of the more complex steps is setting up the material model for the foam (block_50). Select commands for setting up the OMD porous flow model are shown below; users should refer to the input file for the full detailed usage.

1. A *general chemistry model* is defined within this material. This particular example includes a mix of 7 solid and gas phase species with various reactions defined:

```
Begin Aria Material pmdifoam_aria_PF_20lbs
  ..
  Begin general chemistry foam_decomp
    Species Names  = FoamA FoamB FoamC Char CO2 LMWO HMWO
    Species Phases = solid_phase solid_phase solid_phase solid_
↪phase gas_phase gas_phase gas_phase
    Species Variable Name = species
    ..
    Begin Reaction FoamA
      ..
    End
    ..
  End
  ..
End
```

2. Within the `pmdifoam_aria_PF_20lbs` material defined above, a solid (porous) and gas phase is defined; the corresponding species for each phases are added to these sub-blocks. For this material, CVFEM will be used with various upwind methods and mass balance equations being invoked. Permeability parameters are defined for the Darcy model used to compute the gas-phase velocity of this material. In addition, a porous_enthalpy model is used to track the evolution in energy with a `gap_conductance_coefficient` defined for contact resistance. A detailed explanation of the governing porous flow system equations are given *here*.

```
Begin Aria Material pmdifoam_aria_PF_20lbs
  ..
  gap_conductance_coefficient = constant H = 10000.0
  ..
  Begin Material Phase solid_phase
    ..
    SPECIES NAMES = FoamA FoamB FoamC Char
    intrinsic_permeability of FoamA = constant  XX=1.0 YY=1.0␣
↪ZZ=1.0
```

(continues on next page)

(continued from previous page)

```
    intrinsic_permeability of FoamB = constant  XX=1.0 YY=1.0␣
↪ZZ=1.0
    intrinsic_permeability of FoamC = constant  XX=1.0 YY=1.0␣
↪ZZ=1.0
    intrinsic_permeability of Char  = constant  XX=1.0 YY=1.0␣
↪ZZ=1.0
    intrinsic_permeability = volume_average
    transported_enthalpy = standard
    enthalpy = mass_average
    ..
  End
  Begin Material Phase gas_phase
    ..
    SPECIES NAMES = CO2 LMWO HMWO
    velocity = darcy  GX=0. GY=-9.81 GZ=0.
    relative_permeability = Constant  K=1.
    ..
    mass balance advective flux of LMWO     = porous_upwind
    mass balance advective flux of HMWO     = porous_upwind
    mass_balance_diffusive_flux of LMWO     = porous
    mass_balance_diffusive_flux of HMWO     = porous
    transported_enthalpy = porous
    enthalpy_advection   = porous_upwind
    heat_conduction      = porous_simplified_diffusive_enthalpy_
↪flux
    ..
  End
  ..
  Begin general chemistry foam_decomp
    ..
  End
  ..
End
```

The other material used here is a stainless steel type material, which has no chemistry defined, and thus will be solved with standard finite elements as opposed to CVFEM. A `gap_conductance_coefficient` parameter is also defined here to close the contact definition between the stainless steel and foam material.

The next step is to set up the coupled equation system to solve the porous flow system. For this example, a monolithic equation system will be created. At the Aria region scope, CVFEM will be invoked to solve for the porous flow equations. This is done with the following:

```
Begin Solution Options
  Begin Porous Flow Options
    Use CVFEM = true
  End
End
```

The following equation system is then defined within the Aria region:

```
Begin Equation System full_system
  ..
  eq mass_balance for pressure in gas_phase on block_50 using q1 with␣
↪lumped_mass adv src
  source for mass_balance in gas_phase on block_50 = lumped_general_chem_
↪mass  mechanism=foam_decomp

  eq porous_enthalpy for temperature in solid_phase on block_50 using q1␣
↪with lumped_mass diff src
  source for porous_enthalpy in solid_phase on block_50 = volumetric_
↪transfer  remote_phase="gas_phase"
  source for porous_enthalpy in solid_phase on block_50 = lumped_general_
↪chem_mass  mechanism=foam_decomp

  eq porous_enthalpy for temperature in gas_phase on block_50 using q1␣
↪with lumped_mass adv diff src
  source for porous_enthalpy in gas_phase on block_50 = compressible_
↪pressure
  source for porous_enthalpy in gas_phase on block_50 = volumetric_
↪transfer  remote_phase="solid_phase"
  source for porous_enthalpy in gas_phase on block_50 = lumped_general_
↪chem_mass  mechanism=foam_decomp

  eq mass_balance for density in solid_phase on block_50 using q1 with␣
↪lumped_mass src
  source for mass_balance in solid_phase on block_50 = lumped_general_
↪chem_mass  mechanism=foam_decomp

  eq mass_balance for mass_fraction of foama in solid_phase on block_50␣
↪using q1 with lumped_mass src
  source for mass_balance of foama in solid_phase on block_50 = lumped_
↪general_chem_mass  mechanism=foam_decomp

  eq mass_balance for mass_fraction of foamb in solid_phase on block_50␣
↪using q1 with lumped_mass src
```

(continues on next page)

(continued from previous page)

```
  source for mass_balance of foamb in solid_phase on block_50 = lumped_
↪general_chem_mass  mechanism=foam_decomp

  eq mass_balance for mass_fraction of foamc in solid_phase on block_50␣
↪using q1 with lumped_mass src
  source for mass_balance of foamc in solid_phase on block_50 = lumped_
↪general_chem_mass  mechanism=foam_decomp

  eq mass_balance for mass_fraction of char in solid_phase on block_50␣
↪using q1 with fracbal

  eq mass_balance for mass_fraction of lmwo in gas_phase on block_50␣
↪using q1 with lumped_mass adv diff src
  source for mass_balance of lmwo in gas_phase on block_50 = lumped_
↪general_chem_mass  mechanism=foam_decomp

  eq mass_balance for mass_fraction of hmwo in gas_phase on block_50␣
↪using q1 with lumped_mass adv diff src
  source for mass_balance of hmwo in gas_phase on block_50 = lumped_
↪general_chem_mass  mechanism=foam_decomp

  eq mass_balance for mass_fraction of co2 in gas_phase on block_50 using␣
↪q1 with fracbal

  eq energy for temperature on canister using q1 with mass diff
  ..
End
```

The mathematical forms for each equation, their source terms and the derived quantities that are being solved for can be found at *Organic Material Decomposition (OMD)*. The mass_balance, enthalpy and species equations leverages the general_chemistry `foam_decomp` model as a source therm with the `lumped_chemistry` model implementation.

For this example problem, a time-varying dirichlet boundary condition is specified on `heated_surface` to induce the chemical decomposition process. A `user_function` is defined as follows:

```
BEGIN DEFINITION FOR FUNCTION upright_150_rate
   type is piecewise linear
   begin values
     #t(s)     T (K)
     0  303.42
     10  303.44
```

(continues on next page)

(continued from previous page)

```
    ..
    960  1083.06
  end values
END DEFINITION FOR FUNCTION upright_150_rate
```

This function is then inserted as a dirichlet BC with the following syntax:

```
BC dirichlet FOR temperature ON heated_surface = user_function ␣
↪name=upright_150_rate X=Time
```

Lastly, at the interface between the foam and stainless steel materials, a `contact_resistance` interface BC is defined:

```
BC Interface for energy on porous_solid_interface_both = contact_
↪resistance opposite_material_phase ="solid_phase"
BC Interface for porous_enthalpy in solid_phase on porous_solid_interface_
↪both = contact_resistance
```

CDFEM Examples

The Conformal Decomposition Finite Element Method (CDFEM) is a powerful tool for the solution for moving interface problems. Using an evolving description of the interface location, typically provided via a level set field, CDFEM will automatically adapt the mesh to conform to the interface at each time step. This provides an explicit mesh part representation of a dynamic interface which allows for easy application of boundary conditions and a guaranteed sharp interface resolution.

Sloshing

In this example, we will use CDFEM to solve the classic "sloshing" problem, in which a dense liquid surrounded by much lighter gas is initially released from an unstable configuration and allowed to slosh back and forth in a domain under the influence of gravity.

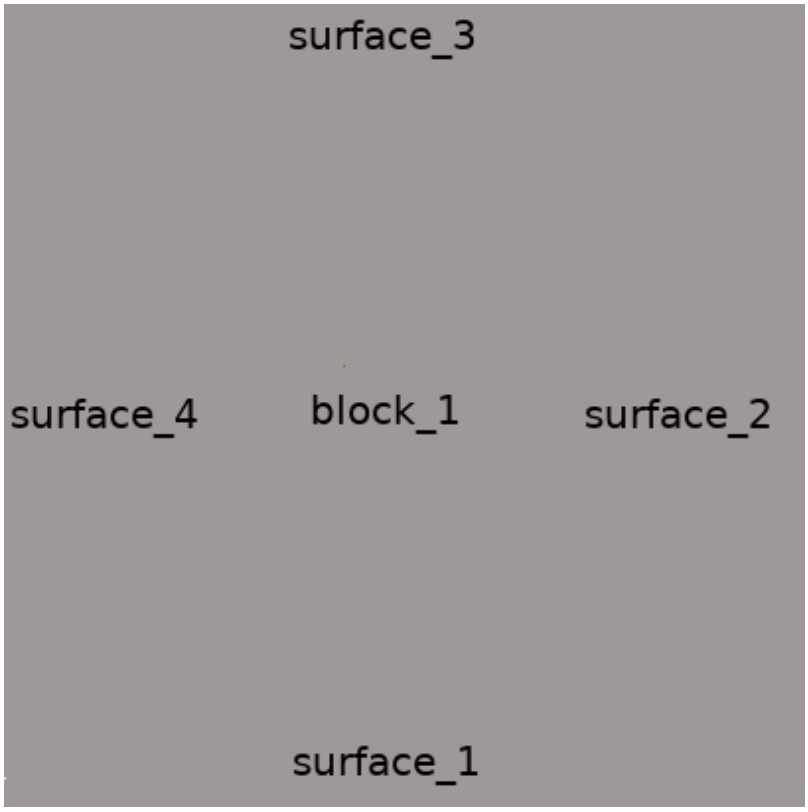
Problem Files

The files required for this example can be downloaded here, or found at `$TESTS/aria_rtest/example/training/cdfem` (where `TESTS=/projects/sierra/tests` in a CEE environment, or your Sierra test distribution otherwise).

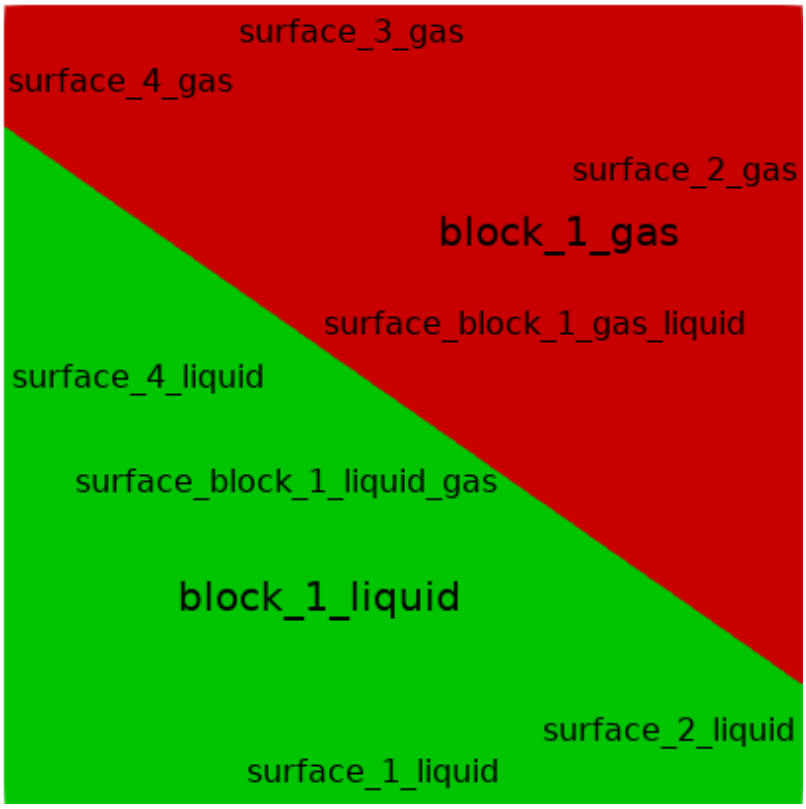
- `Aria input file`
- `Mesh file`

Problem Domain/Mesh

The geometry consists of a single square block with four sidesets, one for each edge of the square. Note at this point there are not separate blocks for the liquid and gas regions. CDFEM will create these blocks when Aria is run.



The objective is to decompose block_1 into liquid and gas phases using CDFEM. The liquid phase will be initially inclined so that it will move under the influence of gravity. All of the sidesets touching block_1 will also be decomposed into the portions of the sideset touching the liquid and gas phases. Finally, two new sidesets will be created at the interface between the phases to allow boundary conditions to be applied to the interface.



Note the naming scheme used for the newly created mesh parts. The new phase blocks are named `[original block name]_[phase name]`. Similarly, the phase-specific sidesets are named `[original sideset name]_[phase name]`. Finally, the interface parts are named `surface_[original block name]_[phase]_[touching phase]`. Also note that sidesets surface_1_gas and surface_3_liquid are not shown. This is because those sidesets exist, but are empty in this initial configuration because each sideset has only one phase touching the entire sideset. If the phases were to move such that the other phase began touching these sidesets, those parts would be populated accordingly.

Aria Input File

The following sections show the commands necessary to set up a two-phase level set CDFEM problem. The input file contains the expected Aria input file components with the addition of some new commands in the finite element model block, a level set equation system, and CDFEM options and level set interface blocks.

Defining Materials

Scope: Domain

You must define a material for each phase in the problem. These materials will be assigned to phase blocks the same way they would be to ordinary mesh blocks, even though the phase blocks do not exist in the input mesh. For this problem, two materials are defined with large differences in density and viscosity to represent the liquid and gas phases. A two phase flow problem like this requires:

- Density
- Viscosity
- A momentum stress model

Additionally, a material will be created to supply a surface tension coefficient at the phase interface. The syntax is shown below:

```
BEGIN ARIA MATERIAL liquid
    Density          = Constant rho=1000
    Viscosity        = constant mu=10.0
    Momentum Stress  = Incompressible_Newtonian
END ARIA MATERIAL liquid
BEGIN ARIA MATERIAL gas
    Density          = Constant rho=1
    Viscosity        = constant mu=0.1
    Momentum Stress  = Incompressible_Newtonian pressure=pressure_1
END ARIA MATERIAL gas
BEGIN ARIA MATERIAL gas_liquid_interface
    Surface Tension = Constant Sigma = 1.
END ARIA MATERIAL gas_liquid_interface
```

Note the specification of the additional `pressure=pressure_1` option for the gas momentum stress. This is because two pressure fields will be used to handle the pressure discontinuity due to surface tension and this option specifies that the gas phase pressure (pressure_1) should be used in the gas phase to calculate the momentum stress instead of the default (pressure), which is the liquid phase pressure.

Linear Solver

Scope: Domain

This problem will use two linear solvers, one for the level set system and one for the coupled momentum/continuity system. The level set system will use the "Scalar_Transport" solver preset (GMRES with SGS preconditioner), following the recommendation for scalar advection diffusion equations in *Solver Selection Guidelines*, with a very tight linear tolerance. The level set system is always linear, but needs to be solved to a very tight tolerance. The most computationally efficient way to accomplish this is to use a linear solver with a tight tolerance and a single nonlinear iteration. The momentum/continuity system will use the "Multiphysics" preset (GMRES with DD-ILU preconditioner), following the recommendation for coupled equations in *Solver Selection Guidelines*.

```
BEGIN TPetra Equation Solver levelset
    BEGIN PRESET SOLVER
        Convergence Tolerance = 1e-12
        Solver Type = scalar_transport
    END
END

BEGIN TPetra Equation Solver momentum
    BEGIN PRESET SOLVER
        Convergence Tolerance = Automatic
        Solver Type = multiphysics
    END
END TPETRA EQUATION SOLVER
```

Define the Finite Element Model

Scope: Domain

The finite element model section is where you indicate which mesh file to use and associate materials with the blocks and surfaces. In this problem, this is also where the phases are specified. The level set interface "LS" is used to divide the liquid and gas phases. The level set is a signed distance field indicating the shortest distance from any point in the domain to the interface. Thus, the interface location is wherever the level set field is zero and the different phases are indicated by the sign of the field.

```
BEGIN FINITE ELEMENT MODEL TwoPhaseFlow
    database name = cdfem_slosh_adapt.g
    coordinate system is cartesian
    Decomposition Method = RCB
```

(continues on next page)

(continued from previous page)

```
    Begin Parameters For Phase gas
        where LS is positive
    End
    Begin Parameters For Phase liquid
        where LS is negative
    End
    Begin Parameters For Block block_1_gas
        Material gas
    End
    Begin Parameters For Block block_1_liquid
        Material liquid
    End
    Begin Parameters For Block block_1
        Material gas
    End
    Begin Parameters For Surface surface_block_1_liquid_gas
        Material gas_liquid_interface
    End
END FINITE ELEMENT MODEL TwoPhaseFlow
```

Solution Control

Scope: Procedure

The solution control block defines how the problem is stepped forward in time. Here, the simulation begin and end times are specified as well as the time stepping scheme. There are two features in this block specific to CDFEM. The first is the `LS_CONSERVED_REDISTANCE` event. The level set field must be periodically redistanced to enforce mass conservation. The recommendation is that this operation is performed whenever the interface moves across a mesh cell. Thus, the redistance operation is conditioned to occur at regular time intervals based on the estimated maximum velocity and mesh size. The second feature is the `Interface Courant Limit` used to control the time step size. Because of the semi-explicit nature of the interface evolution, for best results Courant numbers on the interface should be limited to around 1. Here 0.5 is chosen for finer time resolution. The predictor-corrector tolerance is also set very high, effectively disabling predictor-corrector time step control and relying instead on the interface Courant limit.

```
Begin Solution Control Description
    Use System Main
    Begin System Main
        Simulation Start Time            = 0.0
        Simulation Termination Time      = 2.0
        Begin Transient Stepper
```

(continues on next page)

(continued from previous page)

```
            Advance MyRegion
            Event LS_CONSERVED_REDISTANCE when "(CURRENT_TIME - LAST_LS_
↪CONSERVED_REDISTANCE_TIME) >= 0.01"
        End
    End
    Begin Parameters For Transient Stepper
        Start Time      = 0.0
        Begin Parameters For Aria Region MyRegion
            Initial Time Step Size = 0.005
            Maximum Time Step Size = 0.1
            Minimum Time Step Size = 0.005
            Time Step Variation = Adaptive
            Interface Courant Limit = 0.5
            Predictor-Corrector Tolerance = 1.0
        End
    End
End
```

Aria Region

Scope: Procedure

Two blocks are required in the Aria Region in order to set up a CDFEM problem: the Level Set Interface block and the CDFEM Options block. The Level Set Interface block sets up the level set interface for use in the finite element model block to determine the phase decomposition. A required component is specifying the mesh field to use as the level set distance variable. In this problem, the solution field from the level set equation system is used. Optionally, initial conditions for the level set field can be specified here. In this case, the interface is initialized with a plane centered at the origin and whose normal is provided. Since we are using a level set field that is also a solution variable, initial conditions for it can also be specified in the level set equation system block using the methods documented in *Initial Conditions*.

The CDFEM options block allows settings specific to CDFEM to be specified. If only default CDFEM settings are to be used, this block can be omitted. For this problem, one level of non-conformal adaptivity is used. This means that, prior to modifying the mesh to align it with the interface, cells that are intersected by the interface are refined one time.

Additional options set in the Aria Region scope are the creation of some mesh groups to clarify mesh part names and disabling the use of the two pressure fields in the predictor-corrector time step control scheme. Since the continuity equation does not have an unsteady term, it is possible for pressure to change rapidly with time, potentially driving time step sizes unnecessarily low.

```
BEGIN ARIA REGION MyRegion

    BEGIN LEVEL SET INTERFACE LS
        Distance Variable = solution->LEVEL_SET
        Begin analytic initial condition myIC
            plane normal = 0.7 1. 0 offset = 0
        End
    END

    Predictor Fields = Not Pressure
    Predictor Fields = Not Pressure_1

    Begin CDFEM OPTIONS
        cdfem nonconformal adaptivity levels = 1
    End

    USE FINITE ELEMENT MODEL TwoPhaseFlow
    MESH GROUP gas_blocks = block_1_gas
    MESH GROUP liquid_blocks = block_1_liquid
    MESH GROUP liquid_gas_interface = surface_block_1_liquid_gas
    MESH GROUP nonconformal_blocks = block_1

    # other region commands
END
```

Level Set Equation System

Scope: Aria Region

The level set equation system will govern the advection of the level set field by the fluids and thus define the evolution of the phase interface. In this block, the use of the level set-specific linear solver is specified as well as a very large correction tolerance. Since the level set equation is always linear in the level set variable, this causes the solution to be accepted after only one nonlinear iteration. The Level_Set equation is specified on both the conformal and non-conformal blocks, although the non-conformal blocks only contain the mass term. This is the only equation that should be specified on the non-conformal block.

```
begin equation system LS
    Use Linear Solver levelset

    Nonlinear Solution Strategy    = Newton
    Maximum Nonlinear Iterations   = 3
    Nonlinear Residual Tolerance   = 1e-6
```

(continues on next page)

(continued from previous page)

```
    Nonlinear Correction Tolerance = 1e+12 # LINEAR

    EQ Level_Set for Level_Set on nonconformal_blocks Using Q1 with Mass
    EQ Level_Set for Level_Set on gas_blocks          Using Q1 with Mass␣
↪Adv SUPG
    EQ Level_Set for Level_Set on liquid_blocks       Using Q1 with Mass␣
↪Adv SUPG
end equation system
```

Momentum Equation System

Scope: Aria Region

The momentum equation system will govern the evolution of the fluid velocities. The use of the momentum equation specific solver is specified and momentum and continuity equations set up on the conformal blocks. The surface tension of the interface will create a pressure discontinuity, so different pressure variables are used in each of the phases. Surface tension is activated using the capillary flux boundary condition on the liquid interface. Numerical stabilization of the capillary force is also generally required. No-slip conditions are applied to the bottom and side walls of the domain. The top of the domain is implicitly kept as a zero-flux boundary. Finally, gravitational loads are applied to both phases, which will drive motion of the liquid.

```
begin equation system momentum
    Nonlinear Solution Strategy    =  Newton
    Maximum Nonlinear Iterations   =  10
    Nonlinear Residual Tolerance   =  1e-6
    Nonlinear Correction Tolerance =  0

    USE LINEAR SOLVER momentum

    EQ Momentum for Velocity ON gas_blocks   USING Q1 WITH MASS ADV DIFF␣
↪SRC SUPG
    EQ Momentum for Velocity ON liquid_blocks USING Q1 WITH MASS ADV DIFF␣
↪SRC SUPG
    EQ Continuity_1 for Pressure_1 ON gas_blocks   USING Q1 WITH DIV
    EQ Continuity   for Pressure   ON liquid_blocks USING Q1 WITH DIV
    PRESSURE STABILIZATION is pspg_shakib with scaling = 1.0

    BC FluxBP for Momentum on liquid_gas_interface = Capillary
    BC FluxBP for Momentum on liquid_gas_interface = Capillary_
↪stabilization

```

(continues on next page)

(continued from previous page)

```
    BC dirichlet for velocity_y on surface_1_gas = constant value = 0.0
    BC dirichlet for velocity_y on surface_1_liquid = constant value = 0.0
    BC dirichlet for velocity_x on surface_2_gas = constant value = 0.0
    BC dirichlet for velocity_x on surface_2_liquid = constant value = 0.0
    BC dirichlet for velocity_x on surface_4_gas = constant value = 0.0
    BC dirichlet for velocity_x on surface_4_liquid = constant value = 0.0

    Source For Momentum On gas_blocks   = Hydrostatic gx=0 gy=-10
    Source For Momentum On liquid_blocks = Hydrostatic gx=0 gy=-10
end equation system
```

Solution Output

Scope: Aria Region

A Results Output block is defined to control the output of simulation results. In this case, all solution fields are written to an exodus database at each time step.

```
BEGIN RESULTS OUTPUT LABEL SolutionOutput
    DATABASE NAME = cdfem_slosh_adapt.e
    AT STEP 0, INCREMENT = 1
    TITLE Aria Two Phase Sloshing
    NODAL VARIABLES = solution->Level_Set AS F
    NODAL VARIABLES = solution->Pressure   AS Pliquid
    NODAL VARIABLES = solution->Pressure_1 AS Pgas
    NODAL VARIABLES = solution->Velocity AS V
END RESULTS OUTPUT LABEL SolutionOutput
```

Viewing the Results

Results can be visualized in Paraview or EnSight. You will notice that for this case multiple files are created per processor rank instead of the typical one per processor rank. This is because CDFEM is generally modifying the mesh every time step, and a new exodus file will be generated each time the mesh is modified. These files will have a `-s####` suffix appended to the file name specified in the output block (ie `cdfem_slosh_adapt.e-s0002` in this example). Each file may contain just one or multiple time steps, depending on whether the interface moved significantly enough to trigger a CDFEM mesh modification between output times. Both Paraview and EnSight will seamlessly read these file stacks, so no special steps need to be taken by the user to view output from CDFEM runs.

Restart Examples

IC Based Restart

This example shows how to *restart* a problem in Aria using *Results Output* from a previous simulation. It builds on the example case built up in the *Sierra 100* tutorial.

Problem Files

The files required for this example can be downloaded here, or found at `$TESTS/aria_rtest/example/training/fileRestart` (where `TESTS=/projects/sierra/tests` in a CEE environment, or your Sierra test distribution otherwise).

- `Read_File restart`
- `IO Region restart`
- `Transfer restart`
- `Mesh`

Note: In the present examples, the `start_time` and `end_time` are arbitrarily set as aprepro variables. In practice, these will be determined by external factors e.g. timeouts on HPC queues.

Input Deck Modification

File-based restart can be incorporated in multiple ways, depending on the use case

Using File Directly

This is the simplest use case, where a field from an external file is directly copied into a DOF field. Limitations of this approach include

- The meshes must be identical.
- Only a single time-plane can be applied.
- The IC is applied to the entire domain; a specific part cannot be given.

Using an IO Region

While *Using File Directly* provides a less verbose syntax for file-based restart, it is not always sufficient. Using an IO Region relaxes these restrictions, at the expense of requiring a slightly more verbose syntax. Cases where this is necessary include when

- The property must be interpolated from one mesh to another or in time.

- The IC must be applied on only a subset of the mesh.
- Properties are defined on differing topologies (e.g. an element field).

Using Transfer

The main limitation of *Using an IO Region* is that the transfers are setup with default properties that cannot be changed. Using transfer provides the highest level of control on how the IC is applied, but requires the most verbose syntax.

Setup for these options are discussed in the remainder of this section.

Using File Directly

To use the results of a previous simulation directly, one must first modify the existing finite element model block to read in these results instead of the mesh file.

Note: The implication of this is that the mesh must be the same as the previous simulation. This requirement is relaxed when *Using an IO Region*.

```
$-------------------------------------------------
$   Specify mesh name and settings
$-------------------------------------------------
# Scope: Sierra
Begin Finite Element Model FEModel
  Database Name = heat_cond.e # instead of mesh.g

  ...
End   Finite Element Model FEModel
```

Additionally, the original ICs are replaced with *Ic Read_File* commands referencing the respective quantities in the restart results.

```
$-------------------------------------------------
$           Specify initial conditions
$-------------------------------------------------
# Scope: Sierra > Procedure > Aria Region
Ic Read_File Temperature = T # Temperature field name in output file
Ic Read_File Voltage = V     # Voltage field name in output file
```

By default, the last time in the file will be used. This behavior can be overridden by specifying a time to be used. Note that this will only take the nearest time to the requested one. If interpolation in time is required, consider *Using an IO Region*. See *Ic Read_File* for a more detailed description of its functionality.

Using an IO Region

To apply this strategy, the previous results are first read in and used in an *IO Region*. This requires both loading in the results to a finite element model block

```
$-------------------------------------------------
$   Specify mesh name and settings
$-------------------------------------------------
# Scope: Sierra
Begin Finite Element Model FEModel
  ...
End   Finite Element Model FEModel

Begin Finite Element Model reinitModel
  Database Name = heat_cond.e
End   Finite Element Model reinitModel
```

and using it in an *IO Region*

```
$-------------------------------------------------
$   Setup reinitialization region
$-------------------------------------------------
# Scope: Sierra > Procedure
Begin Input_Output Region reinitRegion
  Use Finite Element Model reinitModel
End   Input_Output Region reinitRegion
```

Similar to above, the original ICs are then replaced with the respective *Ic Input_Output_Region* commands

```
$-------------------------------------------------
$           Specify initial conditions
$-------------------------------------------------
# Scope: Sierra > Procedure > Aria Region
IC Input_Output_Region on all_blocks Temperature = T
IC Input_Output_Region on all_blocks Voltage = V
```

Note: If more than one IO Region is present in the simulation, one must be explicitly selected in the `IC Input_Output_Region` line.

Using Transfer

The *Ic Input_Output_Region* automatically creates the transfers required to initialize a given problem. As a result, it must make some implicit decisions about how these transfers are setup (e.g. extrapolation treatment). At the expense of making the input deck more verbose, these settings can be controlled explicitly by creating the transfers yourself. Similar to above, one must first read the results into an IO region via a finite element block

```
# Scope: Sierra
Begin Finite Element Model reinitModel
  Database Name = heat_cond.e
End   Finite Element Model reinitModel
```

```
# Scope: Sierra > Procedure
Begin Input_Output Region reinitRegion
  Use Finite Element Model reinitModel
End   Input_Output Region reinitRegion
```

Unlike above, the IC lines are remove and not changed. In their place, transfer(s) are defined to control how the information is applied to the IC. See *Transfer* for more information on how to set up transfers.

```
$-----------------------------------------------------
$   Setup IC transfers
$-----------------------------------------------------
# Scope: Sierra > Procedure
Begin Transfer icTransfer
  Interpolate Volume Nodes From reinitRegion To AriaRegion

  Send Field T State none To Solution->Temperature State old
  Send Field V State none To Solution->Voltage State old

  #User-specified settings for transfer
  Nodes Outside Region = Truncate
  ...
End   Transfer icTransfer
```

These transfer(s) are then called at initialization of the restarted simulation by calling them in the *Initialization* block

```
# Scope: Sierra > Procedure
Begin Solution Control Description
  #Define initialize block and use it in system
  begin initialize init
```

(continues on next page)

(continued from previous page)

```
    advance reinitRegion
    advance AriaRegion
    transfer icTransfer
  end   initialize init

  ...

  Begin System Main
    use initialize init

    ...
  End   System Main
End   Solution Control Description
```

Warning: Since each region will initialize its DOFs within the `initialize` block, the IC `transfer` must occur **AFTER** to overwrite the default DOF field.

Restart Database

This example shows how to *restart* a problem in Aria using a *Restart Data* block. Unlike *IC Based Restart*, the restart data approach restarts not only the ICs, but recreates the exact state of the problem contained in the database. It also builds on the example case built up in the *Sierra 100* tutorial.

Problem Files

The files required for this example can be downloaded here, or found at `$TESTS/aria_rtest/example/training/dbRestart` (where `TESTS=/projects/sierra/tests` in a CEE environment, or your Sierra test distribution otherwise).

- `Input File`
- `Mesh`

Note: In the present examples, the `end_time` is arbitrarily set as an aprepro variable. In practice, this is determined by external factors e.g. timeouts on HPC queues.

Input Deck Modification

First, we inform Aria that it can handle restart automatically. This means that Aria will search for the input database, and restart to the latest available time if available. Otherwise it will start a new simulation.

```
$--------------------------------------------------
$   Allow Aria to handle restart automatically
$--------------------------------------------------
# Scope: Sierra
restart = auto

# If a specific time of an existing database is required,
# it can be set here in place of automatic restart
#restart time = 300.0 #seconds
```

Alternatively, the restart time can be explicitly set if a restart database already exists.

Most of the time, restart is used when a simulation is terminated before its desired end time. Considering this, we can set the *Exists = Append* behavior in our results output to append the existing database. This yields a continuous dataset across restarts.

```
$--------------------------------------------------
$           Define contents of output files
$--------------------------------------------------
# Scope: Sierra > Procedure > Aria Region
Begin Results Output AriaOutput
  ...

  Exists = Append
End   Results Output AriaOutput

...

# Scope: Sierra > Procedure > Aria Region
Begin Restart Data RestartOutput
  Input Database Name = heat_cond.rsA
  Output Database Name = heat_cond.rsA

  At Step 0 Increment = 1
  Cycle Count = 5
End   Restart Data RestartOutput
```

Finally, we inform Aria that it must read/write the restart data. Note that there are *various ways* to reduce the overall space taken by these restart databases. In the current example, the database

retains only the last 5 steps.

Note: Each instance of restart will by default append a `.e-sXXXX` extension if necessary to generate a unique file name. In this way, previous restart databases will not be overwritten.

MPMD Examples

PMR - Radiation

This example demonstrates a radiation transport problem using Sierra/PMR for an external PMR solve coupled via MPMD. See *MPMD Radiation* for more discussion on how to setup the Aria and PMR inputs.

Problem Files

The files required for this example can be downloaded here, or found at `$TESTS/aria_rtest/example/training/pmr` (where `TESTS=/projects/sierra/tests` in a CEE environment, or your Sierra test distribution otherwise).

- `Aria input file`
- `PMR input file`
- `Aria mesh file`
- `PMR mesh file`

Input Deck Modification

To introduce an external PMR solve, an MPMD coupling block is first introduced in the Aria input deck. Here, the PMR solve is repeated every 2nd nonlinear Aria iteration.

```
# Scope: Sierra > Procedure > Aria Region
Begin MPMD Coupling
  Coupling Type is Radiation
  Begin Radiation Coupling
    Iterations per Radiation Solve is 2
  End Radiation Coupling
End MPMD Coupling
```

This result is communicated to the Aria region by declaring an `XFER` equation (see *Equation Specification*) for `Radiation_Transport`

```
# Scope: Sierra > Procedure > Aria Region
EQ Radiation_Transport For rte_scalar_flux On Block_1 Using Q1 With xfer
```

This result is then fed into the temperature solve via a SRC

```
# Scope: Sierra > Procedure > Aria Region
EQ Energy For Temperature On block_1 Using Q1 With Diff Src
Source For Energy On block_1 = RTE_FIELD
```

In order to calculate the boundary intensity field sent to Sierra/PMR, the `Emissivity`, `Transmissivity`, and `BC Environment Temperature` must be defined on all boundary surfaces using surface materials

```
# Scope: Sierra
Begin Aria Material wall_a
  emissivity                   = Constant e = 0.6
  transmissivity               = Constant t = 0.4
  bc environment temperature   = Constant T = 1000.0
End

Begin Aria Material wall_b
  emissivity                   = Constant e = 0.6
  transmissivity               = Constant t = 0.4
  irradiation                  = FROM_MPMD_FIELD
  bc environment temperature   = Constant T = 0.0
End
```

Note in this example, the irradiation is also solved from the resulting MPMD solution so that an additional radiation BC can be applied to the surface

```
BC Flux for Energy on surface_2 = Generalized_Rad
```

The reader is referred to the PMR user manual for details on how to modify the provided PMR input deck.

The problem can then be launched with the following command

```
$ mpirun -np 4 aria --mpmd_master --mpi_color 9999 \
    -i aria.i : -np 2 pmr -i pmr.i
```

Teko Examples

Teko Solver for OMD with Porous Flows Example

This example shows how to set up a Teko solver for an OMD example problem with porous flows.

Problem Files

The files required for this example can be downloaded here, or found at `$TESTS/aria_rtest/example/training/omd_fic_enclosure` (where `TESTS=/projects/sierra/tests` in a CEE environment, or your Sierra test distribution otherwise).

- `Input File`
- `Mesh`

This example problem simulates a two dimensional foam-in-a-can experimental setup with enclosure radiation. There are three materials in this example: a fluid-only region (block 6), foam - a porous medium containing both fluid and solid phases (block 5), and a solid material for the can (blocks 1-4). For this particular case, a single Aria region encompassing the entire domain is used. A single, monolithic equation system is used for this simulation. Our main concern in this example, therefore, is demonstrating how to construct a Teko block preconditioner using the advice from the *Road-map for Constructing Teko Solvers* for the monolithic, multiphysics system.

Setting up a Teko Solver

Here we follow steps 1-4 as outlined in the *Road-map for Constructing Teko Solvers*.

1. We start by gather the DOFs in the monolithic linear system, which in alphabetical order are:

```
ENTHALPY
GAS_PHASE_MASS_FRACTION_CO2
GAS_PHASE_MASS_FRACTION_Organic
GAS_PHASE_PRESSURE
GAS_PHASE_TEMPERATURE
MASS_FRACTION_CO2
MASS_FRACTION_Organic
PRESSURE
SOLID_PHASE_DENSITY
SOLID_PHASE_MASS_FRACTION_FoamA
SOLID_PHASE_MASS_FRACTION_FoamB
```

(continues on next page)

(continued from previous page)

```
SOLID_PHASE_TEMPERATURE
TEMPERATURE
VELOCITY
```

2. With the DOFs identified above, we start by assigning each DOF to its own block. We employ a *Block Gauss-Seidel* approach with a direct solver for each diagonal sub-block, as recommended in step 2. The sample input deck block is shown below:

```
begin tpetra equation solver gmres_teko
  begin gmres solver
    convergence tolerance = 1e-8
    begin teko preconditioner
      define matrix subblock number=1  dof=enthalpy
      define matrix subblock number=2  dof=gas_phase_mass_
↪fraction_co2
      define matrix subblock number=3  dof=gas_phase_mass_
↪fraction_organic
      define matrix subblock number=4  dof=gas_phase_pressure
      define matrix subblock number=5  dof=gas_phase_temperature
      define matrix subblock number=6  dof=mass_fraction_co2
      define matrix subblock number=7  dof=mass_fraction_organic
      define matrix subblock number=8  dof=pressure
      define matrix subblock number=9  dof=solid_phase_density
      define matrix subblock number=10 dof=solid_phase_mass_
↪fraction_foama
      define matrix subblock number=11 dof=solid_phase_mass_
↪fraction_foamb
      define matrix subblock number=12 dof=solid_phase_
↪temperature
      define matrix subblock number=13 dof=temperature
      define matrix subblock number=14 dof=velocity

      begin teko subblock solver direct_solver
        begin klu2 solver
        end
      end

      begin block gauss-seidel solver
        use default inverse direct_solver for subblocks
      end
    end
  end
```

(continues on next page)

(continued from previous page)

```
end tpetra equation solver
```

3. Now we need to identify the DOF-to-block mapping. We start with the one-to-one mapping from step 2 and run a single time-step and observe the solve:

```
Equation System all->main:
 * Step  : Transient, Strategy: LINE_SEARCH, Time: 1.00e-04,␣
↪Step: 1.00e-04
 * Matrix: Solver: "gmres_teko", Unknowns: 12270, Nonzeros:␣
↪688594
 * Mesh  : Processor 0 of 1: 2396 of 2396 elems, 2514 of 2514␣
↪nodes
 * Computing View Factors for enclosure encl_Enc1
      N O N L I N E A R              L I N E A R             T I␣
↪M E
------------------------------  -------------------- -----------
↪------------
Step SubItns  Resid    Delta     Itns Status  Resid     Asm/Slv/
↪SubItn Time
---- -------  -------- --------  ---- ------ -------- -----------
↪------------
   1       1 2.16e+02 1.49e+01     93       ok 9.94e-09 7.2e-02/
↪1.2e+00/2.8e-02
   2       1 7.13e-03 4.67e-01    274       ok 9.44e-09 6.4e-02/
↪1.8e+00/2.8e-02
   3       1 4.46e-06 1.68e-05    149       ok 9.05e-09 6.4e-02/
↪1.4e+00/2.2e-02
   4       0 2.20e-06 1.21e-09    128       ok 9.79e-09 6.2e-02/
↪1.3e+00/2.9e-04
Termination reason: 1.20724e-09 < nonlinear_correction_
↪tolerance(1e-06)
```

Considering that we are employing a direct solver for each of the sub-blocks, these iterations counts are too high. Keeping usage of our direct solver, we can start grouping different DOFs to the same sub-block. Through some trial-and-error and physical intuition, we realize that:

a. The enthalpy and pressure equations are strongly coupled

b. The solid phase and gas phase temperatures, along with the gas phase pressure, are strongly coupled

c. All other blocks, in comparison, are only 'weakly' coupled

This is incorporated into the following input deck:

```
begin tpetra equation solver gmres_teko
  begin gmres solver
    convergence tolerance = 1e-8
    begin teko preconditioner
      define matrix subblock number=1  dof=pressure
      define matrix subblock number=1  dof=enthalpy

      define matrix subblock number=2  dof=solid_phase_
↪temperature
      define matrix subblock number=2  dof=gas_phase_temperature
      define matrix subblock number=2  dof=gas_phase_pressure

      define matrix subblock number=3  dof=velocity
      define matrix subblock number=4  dof=temperature
      define matrix subblock number=5  dof=gas_phase_mass_
↪fraction_co2
      define matrix subblock number=6  dof=gas_phase_mass_
↪fraction_organic
      define matrix subblock number=7  dof=mass_fraction_co2
      define matrix subblock number=8  dof=mass_fraction_organic
      define matrix subblock number=9  dof=solid_phase_density
      define matrix subblock number=10 dof=solid_phase_mass_
↪fraction_foama
      define matrix subblock number=11 dof=solid_phase_mass_
↪fraction_foamb

      begin teko subblock solver direct_solver
        begin klu2 solver
        end
      end

      begin block gauss-seidel solver
        use default inverse direct_solver for subblocks
      end
    end
  end
end tpetra equation solver
```

Using the input deck from above, we see that the iteration counts are significantly improved:

```
Equation System all->main:
 * Step   : Transient, Strategy: LINE_SEARCH, Time: 1.00e-04,␣
```

(continues on next page)

(continued from previous page)

```
 ↪Step: 1.00e-04
 * Matrix: Solver: "gmres_teko", Unknowns: 12270, Nonzeros:␣
 ↪688594
 * Mesh  : Processor 0 of 1: 2396 of 2396 elems, 2514 of 2514␣
 ↪nodes
 * Computing View Factors for enclosure encl_Enc1
      N O N L I N E A R              L I N E A R             T I␣
 ↪M E
-------------------------------  -------------------- -----------
 ↪------------
Step SubItns  Resid    Delta     Itns Status  Resid     Asm/Slv/
 ↪SubItn Time
---- -------  -------- --------  ---- ------ -------- -----------
 ↪------------
   1       1 2.16e+02 1.48e+01      3       ok 8.72e-09 6.7e-02/
 ↪7.0e-01/2.8e-02
   2       1 7.13e-03 3.35e-04      3       ok 9.55e-09 6.3e-02/
 ↪7.0e-01/2.8e-02
   3       0 2.69e-06 9.04e-09      3       ok 4.19e-09 6.3e-02/
 ↪6.9e-01/2.5e-04
Termination reason: 9.03819e-09 < nonlinear_correction_
 ↪tolerance(1e-06)
```

4. Now that we can reach convergence in a few iterations with Block Gauss-Seidel, we must now try to remove the direct solver usage for each sub-block. Converting each sub-block one-at-a-time, we observe that the solid phase density and all mass fraction terms can be readily solved via Jacobi. That is, the following input deck yields a solver with the same iteration counts as the direct solver for those sub-blocks

```
begin tpetra equation solver gmres_teko
  begin gmres solver
    convergence tolerance = 1e-8
    begin teko preconditioner
      define matrix subblock number=1  dof=pressure
      define matrix subblock number=1  dof=enthalpy

      define matrix subblock number=2  dof=solid_phase_
 ↪temperature
      define matrix subblock number=2  dof=gas_phase_temperature
      define matrix subblock number=2  dof=gas_phase_pressure

      define matrix subblock number=3  dof=velocity
```

(continues on next page)

(continued from previous page)

```
      define matrix subblock number=4  dof=temperature
      define matrix subblock number=5  dof=gas_phase_mass_
↪fraction_co2
      define matrix subblock number=6  dof=gas_phase_mass_
↪fraction_organic
      define matrix subblock number=7  dof=mass_fraction_co2
      define matrix subblock number=8  dof=mass_fraction_organic
      define matrix subblock number=9  dof=solid_phase_density
      define matrix subblock number=10 dof=solid_phase_mass_
↪fraction_foama
      define matrix subblock number=11 dof=solid_phase_mass_
↪fraction_foamb

      begin teko subblock solver direct_solver
        begin klu2 solver
        end
      end

      begin teko subblock preconditioner jacobi_pc
        begin jacobi preconditioner
        end
      end

      begin block gauss-seidel solver
        use default inverse direct_solver for subblocks
        use inverse jacobi_pc for subblock 5
        use inverse jacobi_pc for subblock 6
        use inverse jacobi_pc for subblock 7
        use inverse jacobi_pc for subblock 8
        use inverse jacobi_pc for subblock 9
        use inverse jacobi_pc for subblock 10
        use inverse jacobi_pc for subblock 11
      end
    end
  end
end tpetra equation solver
```

With the following results from the log file:

```
Equation System all->main:
 * Step  : Transient, Strategy: LINE_SEARCH, Time: 1.00e-04,␣
↪Step: 1.00e-04
 * Matrix: Solver: "gmres_teko", Unknowns: 12270, Nonzeros:␣
```

(continues on next page)

(continued from previous page)

```
↪688594
 * Mesh  : Processor 0 of 1: 2396 of 2396 elems, 2514 of 2514␣
↪nodes
 * Computing View Factors for enclosure encl_Enc1
     N O N L I N E A R             L I N E A R             T I␣
↪M E
------------------------------  -------------------- -----------
↪------------
Step SubItns  Resid    Delta     Itns Status  Resid     Asm/Slv/
↪SubItn Time
---- -------  -------- --------  ---- ------ -------- -----------
↪------------
   1       1 2.16e+02 1.48e+01      3       ok 8.72e-09 6.6e-02/
↪6.9e-01/2.7e-02
   2       1 7.13e-03 3.35e-04      3       ok 9.55e-09 6.4e-02/
↪6.9e-01/2.7e-02
   3       0 2.69e-06 9.04e-09      3       ok 4.19e-09 6.3e-02/
↪6.7e-01/2.4e-04
Termination reason: 9.03819e-09 < nonlinear_correction_
↪tolerance(1e-06)
```

When this occurs, you can generally combine all of the blocks using Jacobi into a single, large block solves via Jacobi. This typically leads to the same iteration count and a faster solver time-to-solution. In addition, we assume that DD-ILU can be easily applied to the velocity and temperature equations. The same cannot be said, however, for the combined pressure-enthalpy block and the combined solid/gas phase temperature-gas phase pressure block. Using a diagnostic preconditioner as recommended in the *Road-map for Constructing Teko Solvers*, we observe that those two sub-blocks require either a better preconditioner or a sub-iteration scheme. Here, we choose the latter option. Combining all of these elements together, we construct our Teko preconditioner. The solver from the input file is shown below:

```
begin tpetra equation solver gmres_teko
  begin gmres solver
    convergence tolerance = 1e-8
    begin teko preconditioner
      define matrix subblock number=1  dof=pressure
      define matrix subblock number=1  dof=enthalpy

      define matrix subblock number=2  dof=solid_phase_
↪temperature
      define matrix subblock number=2  dof=gas_phase_temperature
      define matrix subblock number=2  dof=gas_phase_pressure
```

(continues on next page)

(continued from previous page)

```
     define matrix subblock number=3  dof=velocity
     define matrix subblock number=4  dof=temperature

     define matrix subblock number=5 dof=gas_phase_mass_
↪fraction_co2
     define matrix subblock number=5 dof=gas_phase_mass_
↪fraction_organic
     define matrix subblock number=5 dof=mass_fraction_co2
     define matrix subblock number=5 dof=mass_fraction_organic
     define matrix subblock number=5 dof=solid_phase_density
     define matrix subblock number=5 dof=solid_phase_mass_
↪fraction_foama
     define matrix subblock number=5 dof=solid_phase_mass_
↪fraction_foamb

     begin teko subblock solver ddilu_solver
       begin gmres solver
         convergence tolerance = 1e-2
         maximum iterations = 30
         restart iterations = 30
         begin dd-ilu preconditioner
         end
       end
     end

     begin teko subblock preconditioner ddilu_pc
       begin dd-ilu preconditioner
       end
     end

     begin teko subblock preconditioner jacobi_pc
       begin jacobi preconditioner
       end
     end

     begin block gauss-seidel solver
       use inverse ddilu_solver for subblock 1
       use inverse ddilu_solver for subblock 2
       use inverse ddilu_pc     for subblock 3
       use inverse ddilu_pc     for subblock 4
       use inverse jacobi_pc    for subblock 5
```

(continues on next page)

(continued from previous page)

```
      end
    end
  end
end tpetra equation solver
```

Running the first step and inspecting the logfile, we observe that we have a fast, robust block preconditioner:

```
Equation System all->main:
 * Step  : Transient, Strategy: LINE_SEARCH, Time: 1.00e-04,␣
↪Step: 1.00e-04
 * Matrix: Solver: "gmres_teko", Unknowns: 12270, Nonzeros:␣
↪688594
 * Mesh  : Processor 0 of 1: 2396 of 2396 elems, 2514 of 2514␣
↪nodes
 * Computing View Factors for enclosure encl_Enc1
      N O N L I N E A R             L I N E A R             T I␣
↪M E
------------------------------  -------------------- -----------
↪------------
Step SubItns  Resid    Delta     Itns Status  Resid     Asm/Slv/
↪SubItn Time
---- -------  -------- --------  ---- ------ -------- -----------
↪------------
   1       1 2.16e+02 1.48e+01      4       ok 2.65e-09 7.3e-02/
↪3.8e-01/2.3e-02
   2       1 7.13e-03 3.35e-04      4       ok 2.69e-09 6.2e-02/
↪3.6e-01/2.2e-02
   3       0 2.64e-06 8.99e-09      4       ok 4.15e-09 7.0e-02/
↪3.7e-01/2.4e-04
Termination reason: 8.98683e-09 < nonlinear_correction_
↪tolerance(1e-06)
```

Now that we have completed steps 1-4 as outlined in the *Road-map for Constructing Teko Solvers*, we have constructed a good Teko preconditioner for our OMD multiphysics problem.

6 Running & Troubleshooting

6.1 Running Aria

Loading the Sierra Module

Once you have finished setting up an input file, you are ready to run Aria. From a CEE or HPC UNIX environment at Sandia, you can load the sierra module to access the latest release of Aria.

```
$ module load sierra
```

This will load the current release of Sierra. To load other versions of Sierra, you can use the following modules

- `module load sierra/x.x` - Load version `x.x` (e.g. `module load sierra/5.10`)
- `module load sierra/sprint` - Load the latest sprint release (every 3-week release)
- `module load sierra/daily` - Load the daily build of Sierra.

Warning: Using the `sierra/daily` module exposes you to potential bugs and instabilities since it is actively developed. If the nightly Sierra build process fails, the `sierra/daily` executable may not exist, or may be much older than expected.

To see a list of the available Sierra versions (and other useful modules available like `apps/anaconda3` or `apps/matlab`) you can use

```
$ module avail
```

Running Aria Locally

To run a job on a non-queued system (e.g. a CEE blade or a cee compute machine) you can call `launch` or `mpirun`. For example, to run a job on 4 processors, you would use

```
$ module load sierra
$ launch -n 4 aria -i demo.i
```

The `launch` command is usually equivalent to using `mpirun` for local execution, but handles setting required MPI flags when running on some HPC systems.

```
$ module load sierra
$ mpirun -np 4 aria -i demo.i
```

Using the Sierra Script

The `sierra` script included in the sierra modules provides some additional functionality for launching sierra jobs. Its use is similar to the `launch` and `mpirun` commands.

```
$ sierra --np 4 aria -i demo.i
```

By default, the sierra script will perform extra steps that are not necessary for running Aria. These include:

- Reading the input file to find the mesh file and running `decomp` on it beforehand.
- Reading the input file to find the output files and running `epu` on them after the simulation is done.

Aria will automatically decompose your mesh, which is usually faster than manually running `decomp`, and most visualization tools can view decomposed output files so running `epu` to combine them into a single file is usually not needed either. To use the sierra script without invoking those steps, add the `--run` option.

```
$ sierra --run --np 4 aria -i demo.i
```

The sierra script can also be used to build user plugins using the `--make` option. See the sections on *user plugins* and *user subroutines* for more information.

```
$ sierra --make aria -i usersub_demo.i
```

Running Aria on an HPC

The HPC systems at Sandia use slurm to schedule jobs. To run a job on an HPC you need to submit it to the slurm system along with some additional information (listed below) and the system will put your job in the queue and run at some point later in the future.

- The number of compute nodes to run on.
- The number of cores to use per node.
- The wall-clock duration of the job (slurm will kill the job if it's not done by this time limit).
- A WCID for the job based on the project funding it. The WCID is used for tracking purposes and also determines the job priority. Use the WC Tool web site to check your WCIDs or get a new one.
- Which queue to submit to (most HPCs have "batch", "short", and "long" - "batch" is the standard).

Refer to the HPC homepage for details about queue limits, core counts per node, and other useful HPC information for the machine you intend to run on.

If you are using SAW to submit your jobs, it can handle collecting the required information and submitting the job to the queue. If you do not use SAW, you can submit your jobs manually using either a batch script or the sierra script.

For simple job submissions you can use the sierra script directly after logging in to the HPC you want to run on. For example, to run Aria on 360 processors for up to 24 hours you would log in to the HPC and run the following command.

```
$ sierra --run --np 360 aria -i demo.i --account WCID --queue-name batch -
↪-time-limit 24:00:00
```

For more complicated submissions, you may need to prepare a batch script to perform any custom pre-processing steps you need. There are example submission scripts for different platforms in `/projects/samples` at Sandia. An example script to run Aria is shown below.

```
#!/bin/bash
#SBATCH --nodes=10                     # Number of nodes
#SBATCH --ntasks-per-node=36           # Number of cores per node
#SBATCH --time=24:00:00                # Wall clock time (HH:MM:SS)
#SBATCH --account=PUT_YOUR_WCID_HERE   # WC ID
#SBATCH --job-name=test                # Name of job
#SBATCH --partition=batch              # partition/queue name: short or␣
↪batch

nodes=$SLURM_JOB_NUM_NODES
cores=36

# do any pre-processing steps you need

mpiexec --bind-to core --npernode $cores --n $(($cores*$nodes)) aria -i␣
↪demo.i
```

If you saved the above script as `run_aria` then you would submit it to the queue using

```
$ sbatch run_aria
```

You can check on the status of your queued jobs using `squeue -u myusername`. To see an estimate of when they will start, use `squeue -u myusername --start`.

Running MPMD Jobs

The command to run in MPMD mode is different from what is used to run Aria normally. To run Aria cases you simply run a parallel job of Aria using something discussed above e.g.

```
$ mpirun -np 100 aria -i input.i
```

In this case one executable, Aria, is using 100 cores. With MPMD runs you are launching two separate MPI jobs with two different codes that can communicate. An example MPMD launch command would look like

```
$ mpirun -np 100 aria --mpmd_master --mpi_color 9999 \
    -i input.i : -np 100 follower_app [follower_app args...]
```

The `mpmd_master` flag indicates that Aria will be controlling the execution (the "leader") of the simulation while the other application "follows" Aria. Aria can be run as follower by omitting this flag e.g.

```
$ mpirun -np 100 aria --mpi_color 9999 \
    -i input.i : -np 100 leader_app [leader_app args...]
```

Note that the order of the two apps *does not* matter. The `mpi_color` argument specifies an id to label the cores executing Aria with to distinguish them from the cores executing the coupled application. This can be any integer, and must be unique across the coupled apps.

There is no requirement that the two codes use the same number of cores, so depending on the mesh and computational costs you may choose a different allocation per code. For example, if your coupled app is very expensive, you may allocate more cores to it than Aria:

```
$ mpirun -np 50 aria --mpmd_master --mpi_color 9999 \
    -i input.i : -np 150 other_app [other_app args...]
```

Special care must be taken when submitting MPMD jobs on the HPCs or any queued environment. By default, the two MPMD codes cannot share cores so to launch the case above on an HPC you would need to request an allocation of 200 cores. This is unnecessarily wasteful though if the coupled app would not be using its 150 cores while Aria runs, and Aria would not be using its 50 cores while the coupled app. To get around this, you must enable oversubscription. For couples where the two apps run sequentially (never executing at the same time) you can allow them to share resources. To get an allocation of 100 cores and use them all for both codes you must add additional mpi flags:

```
$ mpiexec --oversubscribe --bind-to core:overload-allowed -np 100  \
    aria --mpmd_master --mpi_color 9999 -i input.i : \
    --bind-to core:overload-allowed -np 100 other_app [other_app args...]
```

Keep in mind that the specific command to use can be platform dependent. A more complete

example submission script on an HPC may look like

```
#!/bin/bash

#SBATCH --nodes=10
#SBATCH --time=48:00:00
#SBATCH --account=PUT_YOUR_WCID_HERE
#SBATCH --job-name=aria
#SBATCH --partition=batch

nodes=$SLURM_JOB_NUM_NODES
cores=36

module load sierra
export OMPI_MCA_rmaps_base_oversubscribe=1
mpiexec --oversubscribe                                          \
  --bind-to core:overload-allowed                                \
  --npernode $cores --n $(($cores*$nodes)) aria --mpmd_master    \
  --mpi_color 9999 -i input.i :                                  \
  --bind-to core:overload-allowed                                \
  --npernode $cores --n $(($cores*$nodes)) other_app [other_app args...]
```

Contact sierra-help@sandia.gov if you need more help or encounter issues running MPMD jobs.

6.2 Reading the Aria log file

Every Aria run outputs a log file with details about the simulation progress. By default this file is named based on the input file name (so an input of `aria.i` would produce a log file `aria.log`). This can be overridden when launching Aria by using the `-o` option, as in `aria -i aria.i -o sierra.log`. Log files are overwritten, so a subsequent launch of the same input file will by default override any prior log files.

The following sections go through the different parts of the log file and how to interpret the information shown in each section. If a simulation encounters an error or produces unexpected behavior, the first place you should look is in the log file.

Preamble

The first thing in the log file is a preamble that shows the Sandia legal statement, when and where the job was run, and what version of Aria was used.

```
+----+----+----+----+----+----+----+----+----+----+----+----+----+----+---
↪-+----

                                    Aria

                Coupled multiphysics including Navier-Stokes,
               elasticity, energy transport, species transport,
                 electrostatics;  free and moving boundaries;
                          transient or steady state.

                              Version 5.10.1-0-g1b0c0bcd

                        With coupled mechanics support for
                           Aria - Coupled multiphysics
                 Encore - Solution Verification Analysis Region

                           Sandia National Laboratories
               Albuquerque, New Mexico and Livermore, California

                        Please email questions and comments to
                                sierra-help@sandia.gov


                 Notice: This computer software was prepared by
                 National Technology and Engineering Solutions
                     of Sandia, hereinafter the Contractor,
                     under Contract DE-NA-0003525 with the
                 Department of Energy (DOE).  All rights in the
                computer software are reserved by DOE on behalf
                    of the United States Government and the
                Contractor as provided in the Contract. You are
                  authorized to use this computer software for
                   Governmental purposes but it is not to be
                 released or distributed to the public. NEITHER
                  THE U.S. GOVERNMENT NOR THE CONTRACTOR MAKES
                ANY WARRANTY, EXPRESS OR IMPLIED, OR ASSUMES ANY
                    LIABILITY FOR THE USE OF THIS SOFTWARE.

+----+----+----+----+----+----+----+----+----+----+----+----+----+----+---
```

(continues on next page)

(continued from previous page)

```
↪-+----

        Directory /scratch/aria/demo
       Executable /projects/sierra/linux_rh7/install/5.10.1/bin/aria
            Built Sep 19 2022 19:07:04
    Build Options linux intel-2021.3.0 release
      Run Started Oct  6 2022 15:52:24
             User sierra
     Architecture ceerws123
             Host ceerws123
         Hardware x86_64
          Running Linux
       Processors 1
Simd vector width 2

      Product                Version          Qualifier
-------------------- ---------------------- ----------
                ACME 2.9.0
                Aria 5.10.1-0-g1b0c0bcd
           Chaparral 3.3.1 development      unreleased
              Encore 5.10.1-0-g1b0c0bcd
               Linux 3.10.0-1127.el7.x86_64
    SIERRA Framework 5.10.1-0-g1b0c0bcd
Sandia Toolkit (STK) 5.10.1-0-g1b0c0bcd
            Trilinos 13.5.0
          UtilityLib 5.10.1-0-g1b0c0bcd
       not specified

+----+----+----+----+----+----+----+----+----+----+----+----+----+----+---
↪-+----
```

In this example, aria `5.10.1` was used for this run, which was built on September 19, 2022. The suffix after the version number (`-0-g1b0c0bcd`) gives a SHA for which commit this version of the code was built with. The summary also shows that the job was launched on October 6, 2022 on a ceerws blade using 1 processor in the `/scratch/aria/demo` directory by the `sierra` user.

Input File

The first thing shown after the preamble is an echo of the input file. If there were any aprepro commands in the input file, those are resolved before this is printed so this is a reflection of the actual commands parsed by Aria. This section is also where any parsing errors will be shown, at the line where the error was detected. The log file echo will be immediately after this line

```
+----+----+----+----+----+----+----+----+----+----+----+----+----+----+---
 ↪-+----
Reading aria.i                                                  Oct  6 2022␣
 ↪15:52:28
```

Initialization

The instantiation section of the log file is shown after the following line

```
+----+----+----+----+----+----+----+----+----+----+----+----+----+----+---
 ↪-+----
Instantiating system                                            Oct  6 2022␣
 ↪15:52:28
```

This section is largely diagnostic, showing defaults and sometimes warnings or errors that may encountered during problem setup. Searching this section for the keyword “warning” can sometimes be useful to ensure that nothing unexpected is being assumed.

The procedure initialization comes after the instantiation section, and has information like a list of available variables for output.

```
+----+----+----+----+----+----+----+----+----+----+----+----+----+----+---
 ↪-+----
Initializing procedure                                          Oct  6 2022␣
 ↪15:52:28

Aria Region "myRegion" has the following fields available for output:
  NODE_RANK Fields:
    * solution->PRESSURE
    * solution->PRESSURE_STKFS_OLD
    * solution->TEMPERATURE
    * solution->TEMPERATURE_STKFS_OLD
    * solution->VELOCITY
    * solution->VELOCITY_STKFS_OLD
  FACE_RANK Fields:
    * face_coverage
```

(continues on next page)

(continued from previous page)

```
  ELEMENT_RANK Fields:
    * current_element_volume
    * initial_element_volume
    * volume_change_ratio
  Global Variables:
    * abv_5
    * CFL_MAX
    * CFL_MIN
```

Run-Time Reporting

After the preamble of the log file comes the actual execution of the model, including linear and nonlinear solver information, time stepping, and variable calculation. The information contained in this section can help determine whether the model is stable and if correct results are being obtained. To illustrate this section of the log file, one time step of the run is shown below and will be further explained in the following sections.

```
+----+----+----+----+----+----+----+----+----+----+----+----+----+----+---
 ↪-+----
Minimum Time Step Selection:
        min           max
Time step selection: dt  = 5.287e+00 (Chemistry minimum timestep).
     1.000e-02     5.287e+00     AriaRegion
Transient Time_Block_2: dt = 5.287
------------------------------------------------
Transient Time_Block_2, step 169, time 2.1985e+03, time step 5.2875e+00,␣
 ↪61.06% complete
------------------------------------------------
Advance AriaRegion, time 2198, time step 5.287

Memory Usage: current = 209186816 (199.5 M), high-water-mark = 217268224␣
 ↪(207.2 M)

Equation System AriaRegion->main:
 * Step   : Transient, Strategy: NEWTON, Time: 2.20e+03, Step: 5.29e+00
 * Matrix: Solver: "Solve_Temperature", Unknowns: 1360, Nonzeros: 11578
 * Mesh   : Processor 0 of 1: 1253 of 1253 elems, 1363 of 1363 nodes
 * Computing View Factors for enclosure space
  N O N L I N E A R               L I N E A R
----------------------   ------------------------------------
Step  Resid    Delta     Itns Status  Resid    Asm/Slv Time
```

(continues on next page)

(continued from previous page)

```
---- -------- --------   ---- ------ -------- ---------------
1    6.37e-02 1.10e+01   59    ok    8.23e-07 7.0e-03/2.0e-03
2    1.10e-04 8.18e-03   58    ok    1.19e-09 7.0e-03/2.0e-03
3    3.06e-07 8.81e-05   61    ok    3.68e-12 6.0e-03/2.0e-03
Termination reason: 8.80515e-05 < nonlinear_correction_tolerance(0.001)

Field            min     @ id    max     @ id max-chg  @ id pred-err ␣
↪pred-min  @ id  pred-max  @ id
------------- ---------- ---- ---------- ---- -------- ---- -------- -----
↪----- ---- ---------- ----
* TEMPERATURE 3.8650e+02  412 8.4155e+02  512 3.93e+00 1443 3.53e-04 3.
↪8650e+02  412 8.4155e+02  512

DT_TEMPERATURE : MIN(value,loc) = (-1.18e+00, 1443) : MAX(value,loc) = (9.
↪68e-01, 249)
Segregated solution procedure converged after 1 iterations.
-------------- For equation system name: main --------------
Global predictor error = 3.531e-04
Time step selection: dt <= 2.326e+01 (based on Predictor-Corrector␣
↪Tolerance).
Time step selection: dt <= 1.057e+01 (based on Maximum Time Step Size␣
↪Ratio).
Time step selection: dt <= 6.000e+01 (based on Maximum Time Step Size).
Time step selection: dt  = 5.277e+00 (Minimum Chemistry timestep block_6).
Time step selection: dt <= 1.277e+01 (based on Stability limit for dt␣
↪Ratio).
Time step selection: dt  = 1.057e+01 (Adaptive time stepping result).
Killed  1 elements this timestep, for a total of 87 dead.
        Global Variable               Value
------------------------------- ----------------
Conv_ipo                        189.224
Conv_ipo_surface_block_2_edge2_2 0
Conv_ipo_surface_block_3_edge2_2 0
killed_elements                 1
space_area                      0.155955
Tmax                            386.498 841.553
total_dead_elements             87
Region::execute() time for AriaRegion: 3.537e+00 sec.
```

Current Time information

The first part of the above section deals with the time step and temporal information about the running model. This section can be seen below:

```
+----+----+----+----+----+----+----+----+----+----+----+----+----+----+---
 ↪-+----
Minimum Time Step Selection:
        min           max
Time step selection: dt  = 5.287e+00 (Chemistry minimum timestep).
     1.000e-02     5.287e+00     AriaRegion
Transient Time_Block_2: dt = 5.287
-------------------------------------------------
Transient Time_Block_2, step 169, time 2.1985e+03, time step 5.2875e+00,␣
 ↪61.06% complete
-------------------------------------------------
Advance AriaRegion, time 2198, time step 5.287
```

Aria determined the current time step size based on factors from the previous time step. Here, the maximum allowed time step size is 5.287, which is based on the minimum time step allowed for the chemistry model. It can also be seen that the run is currently in `Time_Block_2` as is set up in Solution Control.

The information contained within the dotted lines allows one to gauge how far along a simulation is. In that one line, one can find the current time block, the iteration number, current simulation time, current time step, and the percent completion of the current time block.

Linear/Nonlinear Solves

The next part of the runtime information deals with the linear and nonlinear solution:

```
Equation System AriaRegion->main:
 * Step   : Transient, Strategy: NEWTON, Time: 2.20e+03, Step: 5.29e+00
 * Matrix: Solver: "Solve_Temperature", Unknowns: 1360, Nonzeros: 11578
 * Mesh   : Processor 0 of 1: 1253 of 1253 elems, 1363 of 1363 nodes
 * Computing View Factors for enclosure space
  N O N L I N E A R                L I N E A R
----------------------   -----------------------------------
Step  Resid    Delta     Itns Status  Resid    Asm/Slv Time
---- -------- --------   ---- ------ -------- ---------------
1    6.37e-02 1.10e+01   59    ok    8.23e-07 7.0e-03/2.0e-03
2    1.10e-04 8.18e-03   58    ok    1.19e-09 7.0e-03/2.0e-03
3    3.06e-07 8.81e-05   61    ok    3.68e-12 6.0e-03/2.0e-03
Termination reason: 8.80515e-05 < nonlinear_correction_tolerance(0.001)
```

(continues on next page)

(continued from previous page)

```
Field            min     @ id    max     @ id max-chg  @ id pred-err ␣
 ↪pred-min  @ id  pred-max  @ id
------------ ---------- ---- ---------- ---- -------- ---- -------- -----
 ↪----- ---- ---------- ----
* TEMPERATURE 3.8650e+02  412 8.4155e+02  512 3.93e+00 1443 3.53e-04 3.
 ↪8650e+02  412 8.4155e+02  512

DT_TEMPERATURE : MIN(value,loc) = (-1.18e+00, 1443) : MAX(value,loc) = (9.
 ↪68e-01, 249)
Segregated solution procedure converged after 1 iterations.
```

Each bullet point here gives some information as to the solution time and strategy, as well as some information about the matrix being solved. The line `Computing View Factors ...` indicates that enclosure radiation viewfactors are being solved.

The meanings of each column of the `NONLINEAR/LINEAR` table are shown in the list below.

Step
: The nonlinear step being taken - here there are three nonlinear steps in the timestep

Resid
: Nonlinear residual at the end of each nonlinear step and before the linear steps are taken

Delta
: Difference in the solution variable(s) compared to the last nonlinear solution

Itns
: Number of linear iterations per nonlinear step

Status
: Whether linear solver completed successfully (output is either *ok* or an error code or message)

Resid
: Linear residual at the end of the linear iterations

Asm/Slv Time
: Linear system assembly/solve times

This section of the log file also shows the reason why the time step exited - usually due to reaching some desired tolerance or failure to converge. Here, the solution was reached when the change in temperature from the previous nonlinear step (`Delta = 8.81e-05`) was less than the nonlinear correction tolerance of 0.001.

The lines below the table also show useful output on the fields being solved - here, for

temperature. We can see that the minimum temperature is 386.5 at node 412, and the maximum temperature is 841.6 at node 512. Information about the predictor-corrector errors are also shown, as well as the time derivative of temperature.

Timestep Determination

Based on the solution at the current step, the timestep is re-calculated. This is shown in the following lines:

```
-------------- For equation system name: main --------------
Global predictor error = 3.531e-04
Time step selection: dt <= 2.326e+01 (based on Predictor-Corrector␣
↪Tolerance).
Time step selection: dt <= 1.057e+01 (based on Maximum Time Step Size␣
↪Ratio).
Time step selection: dt <= 6.000e+01 (based on Maximum Time Step Size).
Time step selection: dt  = 5.277e+00 (Minimum Chemistry timestep block_6).
Time step selection: dt <= 1.277e+01 (based on Stability limit for dt␣
↪Ratio).
Time step selection: dt  = 1.057e+01 (Adaptive time stepping result).
```

These restrictions will change based on the method in which the solution is solved as well as the physics being solved. Here, the timestep may be limited by adaptive time stepping restrictions (predictor-corrector tolerance, maximum timestep ratio, and maximum time step size), chemistry timestep, and BDF2 stability limit. One can see that the Chemistry timestep of 5.28s is the limiting timestep, and this will be the timestep size which is used in the next timestep. See *Time Step Selection* for more information.

Global Variable Output

At the end of each timestep, a summary is given of the global variables in the model. If there are global variables that the user does not wish to output the `GLOBAL VARIABLES EXCLUDED FROM LOG FILE = [list of variables]` input deck line may be used.

```
        Global Variable                Value
------------------------------- ----------------
Conv_ipo                        189.224
Conv_ipo_surface_block_2_edge2_2 0
Conv_ipo_surface_block_3_edge2_2 0
killed_elements                 1
space_area                      0.155955
Tmax                            386.498 841.553
```

(continues on next page)

(continued from previous page)

```
total_dead_elements                87
Region::execute() time for AriaRegion: 3.537e+00 sec.
```

This section is useful for run-time monitoring of key values, as well as how long the execution time was, in order to locate any slow-running parts of the model through time.

Timing Summary

Aria contains many timers which encapsulate certain operations of each simulation. The timing summary shows the results of these timers and are organized so that aspects of the model that take more time can be found. If your simulation is running slower than you expect, or you are looking for ways to speed it up, the timer section can give useful information about what parts of the simulation are most expensive.

A sample timing summary can be seen below and will be further explained:

```
Timing summary of 1 processor
                  Timer                     Count       CPU Time             ␣
 ↪ Wall Time
---------------------------------------- ----- ---------------------- -----
 ↪----------------
Sierra                                         1    15:14.415 (100.0%)    ␣
↪15:19.072 (100.0%)
  Procedure AriaProcedure
    Initialize                                 1        0.017 (<0.01%)     ␣
↪   0.017 (<0.01%)
    Execute                                    1    15:13.010 (99.85%)    ␣
↪15:17.659 (99.85%)
    Mesh input                                 1        0.168 ( 0.02%)     ␣
↪   0.170 ( 0.02%)
    Mesh output                              357        0.905 ( 0.10%)     ␣
↪   1.196 ( 0.13%)
    Region AriaRegion
      Initialize                               1        0.015 (<0.01%)     ␣
↪   0.017 (<0.01%)
      Execute                                357    15:11.364 (99.67%)    ␣
↪15:15.619 (99.62%)
        Nonlinear Iteration                  997       16.482 ( 1.80%)     ␣
↪  16.568 ( 1.80%)
          Preprocessing                      997        6.811 ( 0.74%)     ␣
↪   6.865 ( 0.75%)
            Chemistry Kinetics               997        6.768 ( 0.74%)     ␣
```

(continues on next page)

(continued from previous page)

```
↪  6.820 ( 0.74%)
         LinSys Assembly                997        6.091 ( 0.67%)      ␣
↪  6.119 ( 0.67%)
         LinSys Solve                   997        2.458 ( 0.27%)      ␣
↪  2.472 ( 0.27%)
         LinSys Scatter                 997        0.276 ( 0.03%)      ␣
↪  0.273 ( 0.03%)
         Nonlinear Utilities           3988        0.668 ( 0.07%)      ␣
↪  0.669 ( 0.07%)
       Postprocessing                   358        0.034 (<0.01%)      ␣
↪  0.046 (<0.01%)
       Utilities                       2144    14:51.790 (97.53%)      ␣
↪14:54.651 (97.34%)
         Parallel Sync                 5776        0.012 (<0.01%)      ␣
↪  0.010 (<0.01%)
         Chaparral Viewfactor           357    14:51.546 (97.50%)      ␣
↪14:54.417 (97.32%)
         Chaparral Radiosity            997        0.629 ( 0.07%)      ␣
↪  0.623 ( 0.07%)
       Chemistry Utilities             3062        0.085 (<0.01%)      ␣
↪  0.079 (<0.01%)
       Contact Search                   357        0.003 (<0.01%)      ␣
↪  0.000 (<0.01%)
                                       1074        0.259 ( 0.03%)      ␣
↪  0.372 ( 0.04%)
       Tmax                             358        0.275 ( 0.03%)      ␣
↪  0.277 ( 0.03%)
     Mesh input                           1        0.168 ( 0.02%)      ␣
↪  0.170 ( 0.02%)
     Mesh output                        357        0.904 ( 0.10%)      ␣
↪  1.196 ( 0.13%)
       Results output                   358        0.326 ( 0.04%)      ␣
↪  0.420 ( 0.05%)
       Restart output                   358        0.549 ( 0.06%)      ␣
↪  0.729 ( 0.08%)
       History output                   358        0.015 (<0.01%)      ␣
↪  0.027 (<0.01%)
       Heartbeat output                 358        0.020 (<0.01%)      ␣
↪  0.036 (<0.01%)
       Control                          109        0.001 (<0.01%)      ␣
↪  0.005 (<0.01%)
       Vis output                       358        0.001 (<0.01%)      ␣
```

(continues on next page)

(continued from previous page)

```
↪  0.000 (<0.01%)
 Perf: RunSierra                              1        1.218 ( 0.13%)     ␣
↪  1.225 ( 0.13%)
   Perf: RunSierra::parse                     1        1.218 ( 0.13%)     ␣
↪  1.225 ( 0.13%)
   Perf: RunSierra::Domain::execute           1    15:13.197 (99.87%)    ␣
↪15:17.846 (99.87%)
```

Each entry shows the CPU and Wall Times associated with it, along with a percentage, which is the percentage of the total run time. One can also see where a timer resides with respect to the Procedure, Region, etc.~by the level of indention in the timing summary. For example, one can see by the indention that `Nonlinear Iteration` is a part of `Execute`, which resides in the `Region ariaRegion`. The sum of timers of all entries on a certain indention level should be less than or equal to the timer of the parent entry, since the parent timer is a wrapper around the children timers and the corresponding code execution.

Failed Time Step

In most cases, failure of a step leads to the timestep being reduced so that the step can be tried again. See *Time Step Failure* for more discussion on this. Some common ways the step can fail are included below, with examples of the corresponding log output

Nonlinear Convergence Failure

This failure occurs when the step fails to converge in the permitted number of nonlinear iterations (see *Maximum Nonlinear Iterations*). If this message persists after decreasing the timestep, it can imply an ill-posed problem or material model.

```
Equation System myRegion->main:
* Step   : Transient, Strategy: NEWTON, Time: 5.75e-03, Step: 6.62e-04
* Matrix: Solver: "solve_temperature", Unknowns: 297, Nonzeros: 4753
* Mesh   : Processor 0 of 1: 128 of 128 elems, 297 of 297 nodes
  N O N L I N E A R                 L I N E A R
----------------------   ------------------------------------
Step  Resid    Delta     Itns  Status   Resid    Asm/Slv Time
---- -------- --------   ---- -------- -------- ---------------
1    1.84e+00 1.76e+01   13    ok       9.38e-11 9.7e-04/6.2e-04
2    8.94e-01 8.28e+00   13    ok       9.14e-11 9.6e-04/6.0e-04
3    8.53e-01 7.41e+00   14    ok       1.95e-11 9.5e-04/6.3e-04
4    8.12e-01 7.24e+00   13    ok       2.14e-11 9.5e-04/6.1e-04
5    6.32e-01 6.50e+00   14    ok       5.73e-11 9.5e-04/6.4e-04
6    7.23e-01 5.63e+00   13    ok       3.23e-11 9.5e-04/5.9e-04
7    5.11e-01 3.85e+00   14    ok       2.30e-11 9.5e-04/6.3e-04
```

(continues on next page)

(continued from previous page)

```
8    3.97e-01 1.70e+00   13    ok      5.51e-11 9.5e-04/6.0e-04
9    3.86e-02 1.43e-01   14    ok      2.15e-11 9.5e-04/6.4e-04
10   3.41e-04 1.63e-03   13    ok      9.07e-11 9.5e-04/6.0e-04
reached maximum_nonlinear_iterations(10)
Region::execute() time for myRegion: 1.78e-02 sec.
Transient TimeBlock1 failed, step 56, time 5.0893e-03, time step 6.
↪6206e-04

  Retrying step with a time step corresponding to time step size␣
↪ratio of 0.5
```

Non-finite nonlinear residual norm

This failure occurs when the linear system encounters NaN/Inf entries. If this message persists after decreasing the timestep, it can imply an ill-posed problem or material model.

```
Equation System myRegion->main:
* Step   : Transient, Strategy: NEWTON, Time: 1.00e-01, Step: 1.00e-01
* Matrix: Solver: "solve_temperature_tpetra", Unknowns: 125,␣
↪Nonzeros: 2197
* Mesh   : Processor 0 of 2: 48 of 64 elems, 100 of 125 nodes
Region::execute() time for myRegion: 5.82e-03 sec.
    1.000e-16     5.000e-02     myRegion
Transient MySolveBlock failed, step 1, time 0.0000e+00, time step 1.
↪0000e-01
ERROR: Non-finite nonlinear residual norm.
  Retrying step with a time step corresponding to time step size␣
↪ratio of 0.5
```

Linear Solver failure

This failure mode happens when the linear solver fails to solve the nonlinear iteration. If these issues persist, see *Linear Solver Failure* for guidance on troubleshooting these errors.

```
Equation System The_Region->LENS:
* Step   : Transient, Strategy: NEWTON, Time: 5.40e-02, Step: 3.67e-04
* Matrix: Solver: "gmres_ilut", Unknowns: 6247772, Nonzeros:␣
↪467738778
* Mesh   : Processor 0 of 832: 13913 of 7441357 elems, 2874 of␣
↪1243618 nodes
  N O N L I N E A R               L I N E A R
----------------------   ------------------------------------
Step  Resid    Delta     Itns Status  Resid    Asm/Slv Time
---- -------- --------   ---- ------ -------- ---------------
  1 8.02e+00 3.62e+08    165     ok 7.92e-06 9.4e-01/7.9e+00
```

(continues on next page)

(continued from previous page)

```
  2 3.53e+00 4.09e+08    224     ok 3.47e-06 9.3e-01/9.9e+00
  3 3.67e+06 1.53e+15    184     ok 3.62e+00 9.3e-01/8.5e+00
ERROR: Linear system failed due to excessive residual ratio growth.
Region::execute() time for The_Region: 3.63e+01 sec.
Transient The_Time_Block_1 failed, step 112, time 5.3627e-02, time␣
↪step 3.6750e-04

  Retrying step with a time step corresponding to time step size␣
↪ratio of 0.8
```

Chemistry/Radiation/Generic submodel failures

This failure occurs when one of the submodels' solves fails, e.g. the chemeq solver failure below

```
Equation System The_Region->energy:
* Step  : Transient, Strategy: NEWTON, Time: 1.30e+02, Step: 5.16e-02
* Matrix: Solver: "Iterative_Solver", Unknowns: 1589, Nonzeros: 63729
* Mesh  : Processor 0 of 4: 123 of 443 elems, 289 of 987 nodes
  N O N L I N E A R                L I N E A R
----------------------   ---------------------------------------
Step  Resid    Delta     Itns  Status   Resid    Asm/Slv Time
---- -------- --------   ---- -------- -------- ---------------
  1 1.18e+03 9.83e+02      6       ok 1.87e-08 9.3e-03/8.0e-03
  2 1.71e+03 1.42e+03      6       ok 2.85e-08 9.7e-03/7.8e-03
  3 6.28e+02 5.11e+02      6       ok 2.36e-08 1.5e-02/7.6e-03
  4 7.14e+02 5.76e+02      6       ok 4.13e-09 1.7e-02/7.6e-03
  5 8.67e+02 6.20e+02      6       ok 1.38e-08 2.1e-02/7.7e-03
  6 1.41e+03 9.65e+02      6       ok 7.46e-08 2.5e-02/7.6e-03
  7 1.37e+03 9.42e+02      6       ok 6.65e-08 3.0e-02/7.6e-03
  8 9.76e+02 7.00e+02      6       ok 7.03e-08 3.5e-02/7.6e-03
  9 8.30e+02 6.20e+02      6       ok 6.50e-08 3.8e-02/7.6e-03
  10 2.40e+02 2.06e+02      5       ok 3.36e-08 4.1e-02/7.4e-03
  11 7.78e+02 6.09e+02      4       ok 9.48e-08 4.2e-02/6.9e-03
  12 6.53e+02 4.70e+02      4       ok 2.54e-07 4.2e-02/6.9e-03
ChemEq ODE in material ChemeqEM model ChemeqEM on block_5:
  chemeq exited with nonzero exit status ... failed to converge.
  CVode error test failure in CVode
Species concentrations before ODE solve:
  h2oa = 0.0622659
  h2og = 7.86947e-05
  tatb = 2.89618
  mf = 6.31361e-13
  h2om = 6.31361e-13
```

(continues on next page)

(continued from previous page)

```
  gasm = 2.44478e-25
  carbonm = 1.56736e-25
  gast = 1.2128e-11
  carbont = 6.30655e-12
Phase Pressure before ODE solve:  8070.71
Temperature before ODE solve:  1940.37
Rate Multiplier before ODE solve:  1
Region::execute() time for The_Region: 4.63e-01 sec.
Transient TimeBlock failed, step 350, time 1.30217e+02, time step 5.
↪1644e-02
ERROR: Chemistry failure - retrying time step
  Retrying step with a time step corresponding to time step size␣
↪ratio of 0.5
```

Missed Phenomenon

One may also find that a timestep is rejected because some underlying physical process was not captured, e.g.

```
Equation System The_Region->energy:
* Step  : Transient, Strategy: NEWTON, Time: 1.20e+02, Step: 1.00e+00
* Matrix: Solver: "Iterative_Solver", Unknowns: 1589, Nonzeros: 63729
* Mesh  : Processor 0 of 4: 123 of 443 elems, 289 of 987 nodes
  N O N L I N E A R                 L I N E A R
----------------------   ------------------------------------
Step  Resid    Delta     Itns  Status   Resid    Asm/Slv Time
---- -------- --------   ---- -------- -------- ---------------
  1 5.16e+01 7.24e+02     14       ok 6.71e-08 1.0e-02/1.1e-02
  2 2.01e-05 2.30e+00      5       ok 2.27e-10 9.2e-03/7.4e-03
  3 8.59e-09                     NoOp          9.3e-03
Termination reason: 8.59282e-09 < nonlinear_residual_tolerance(5e-
↪07),
                and 0 < nonlinear_correction_tolerance(1e-09)

Field               min     @ id     max     @ id  max-chg  @ id ␣
↪pred-err  pred-min  @ id   pred-max  @ id
---------------- ---------- ----- ---------- ----- -------- ----- ---
↪----- ---------- ----- ---------- -----
  DENSITY        3.5813e+01 27005 5.1571e+01 27006 1.89e+00 27005 3.
↪10e-02 3.3922e+01 27005 5.0338e+01 27006
  SPECIES        5.1795e-01 27005 7.6904e-01 27006 3.96e-02 27006 4.
↪49e-02 4.8946e-01 27005 7.2940e-01 27006
  SPECIES_carbon 2.9143e-13 26393 1.4224e+00 26350 1.42e+00 26269 1.
↪01e-01 2.8892e-13 26393 7.0805e-03 26350
```

(continues on next page)

(continued from previous page)

```
  SPECIES_em     1.1284e+00 26350 3.0000e+00 26306 1.86e+00 26350 6.
 ↪32e-02 2.9907e+00 26350 3.0000e+00 26306
  SPECIES_gasses 3.8347e-12 26393 1.8716e+01 26350 1.86e+01 26350 1.
 ↪01e-01 3.8016e-12 26393 9.3164e-02 26350
* TEMPERATURE    3.0013e+02 26308 1.2000e+03 26818 4.57e+02 26278 1.
 ↪92e-02 3.0013e+02 26308 1.2000e+03 26818

Region::execute() time for The_Region: 7.35e-02 sec.
Transient TimeBlock failed, step 120, time 1.1900e+02, time step 1.
 ↪0000e+00
ERROR: Missed Chemistry Deactivation.
Retrying step with a time step size ratio of 0.655535
```

6.3 Reading the Chaparral log file

When enclosure radiation is defined and set up in the input file, an external library called Chaparral is utilized. Chaparral output is sent to a separate log file from the main Aria log file (usually named with `.chap.log`). Chaparral will output useful information regarding the viewfactor calculation, matrix smoothing, and the radiosity solver. Each section below will describe the output one may expect from Chaparral.

Instantiation

This output only occurs at the very start of the Chaparral output, but can help a user remember the number of processors used and the number of enclosures in the model:

```
*****************************************************************
  C H A P A R R A L  --  Version 3.3.1 development  --  unreleased
*****************************************************************

Initializing for:
    number of processors  = 1
    number of enclosures  = 1
    max number of patches = 1000
```

Viewfactor Calculation

At the beginning of the viewfactor calculation, a banner will be displayed which shows the current simulation step and time, which is useful in correlating the Chaparral output to Aria output, as well as an estimate of the rowsum for each enclosure:

```
************************************************************************
C O M P U T E    V I E W F A C T O R S
Step = 214   Time = 2425.53
************************************************************************

************************************************************************
V I E W F A C T O R    C A L C U L A T I O N
************************************************************************

Calculating viewfactors for enclosure <space>
  enclosure geometry:    axisymmetric
  enclosure type:        partial (area=100), blocking
  # of rotations:        16
  # of patches:          62
  # of facets:           976
  # of nodes:            63
  spatial tolerance:     1e-06
  BSP target max depth:  10
  BSP target min length: 50
  BSP Tree num leafs:    28
  BSP Tree max depth:    5
  BSP Tree min length:   27
  BSP Tree max length:   45
  output level:          2

Data segment memory size = 0.00Mb

Calling VF_Hemicube()...
  resolution       = 400
  max subdivisions = 4
  min separation   = 5

Minimum Partial Enclosure Area = 0.151296

  Minimum effective surface radius    = 0.00916667
  Minimum separation distance         = 0.00558985
  Maximum desired surface subdivision = 9, 46
  Actual maximum surface subdivision  = 4
```

(continues on next page)

(continued from previous page)

```
Data segment memory size = 0.00Mb

Elapsed time = 3.92
```

While some of this output is merely a summary of user-specified or default tolerances, some information is useful during the simulation or for debugging. First, the user-defined name of the enclosure is shown in < > brackets (i.e., `space`). The number of facets and nodes that participate in the enclosure can also be useful in determining if the enclosure is complete in some instances. Last, if one is specifying a partial enclosure, then the `Minimum Partial Enclosure Area` is useful in correct specification of the corresponding parameter in the input file.

For simulations in which the enclosure does not change through time, the viewfactor calculation and corresponding text output will only appear once per enclosure after instantiation.

Viewfactor Smoothing

If viewfactor smoothing is specified, information about this step is also output. An example is shown below:

```
*****************************************************************
V I E W F A C T O R    M A T R I X    S M O O T H I N G
*****************************************************************

Smoothing of viewfactor matrix for enclosure <space>
  PCG Solver, wt = 2.0
  Max iterations = 500
  Tolerance      = 1e-07

  Enforcing reciprocity by addition...
    Nonzero lower triangular entries = 688 changed to 688
    Nonzero upper triangular entries = 687 changed to 688
    Elapsed time                     = 0.00
  Number of passes     = 1
  Number of iterations = 26
  Elapsed time         = 0.00
```

Again, the enclosure name is denoted within the < > brackets, with another summary of defined values. This section is useful for debugging problematic enclosures, since viewfactor smoothing can fail when the enclosure is ill-defined.

Viewfactor Summary

Once the viewfactors have been computed and smoothing has been applied, Chaparral will output the summary of how well the operations have done. This is shown in the `Viewfactor Matrix Summary`:

```
*****************************************************************
V I E W F A C T O R    M A T R I X    S U M M A R Y
*****************************************************************

Viewfactor matrix summary for enclosure <space>

Target rowsum                  = 62
Raw rowsum total               = 62.000000
Raw rowsum error min           = -1.1479e-07
Raw rowsum error max           =  1.1025e-07
Raw rowsum error mean          =  4.9383e-09 +/- 4.9314e-08
Smoothed rowsum total          = 62.000000
Smoothed rowsum error min      = -7.0897e-08
Smoothed rowsum error max      =  5.9255e-08
Smoothed rowsum error mean     =  2.8875e-10 +/- 2.7300e-14

Viewfactor matrix is 36.19% dense

Total elapsed time = 3.93 (sec)
  (Initialization) = 0.01
  (Calculation)    = 3.92
  (Smoothing)      = 0.00

Data segment memory size  = 0.00Mb
```

Here one can see that for the `space` enclosure, the computed rowsum closely matched the target rowsum, with and without smoothing. If one is also concerned about timing, an itemized timing summary is given for the viewfactor matrix setup. This is probably the most useful section to look at if one is concerned about issues with certain enclosures, as well as judging the cost and usefulness of viewfactor smoothing.

Radiosity Solution

After the viewfactor matrix has been populated, Chaparral will use this to compute the radiosity matrix. Output for one iteration of the radiosity matrix solver is shown below:

```
*****************************************************************
R A D I O S I T Y    S O L V E S
Step = 214   Time = 2425.53 Time_Step = 6.18025 Iteration = 1
*****************************************************************



*********************************************************************
R A D I O S I T Y    M A T R I X    S O L V E R
*********************************************************************

Solving radiosity equations with GMRES for enclosureID <space>

Initial residual = 1.553372e+03    tol = 2.000000e-07
    iter:    0   residual = 1.553372e+03
    iter:    1   residual = 2.657621e+04
    iter:    2   residual = 6.553812e+03
    iter:    3   residual = 1.226445e+03
    iter:    4   residual = 3.526103e+02
    iter:    5   residual = 8.437261e+01
    iter:    6   residual = 3.212877e+01
    iter:    7   residual = 8.262992e+00
    iter:    8   residual = 2.397060e+00
    iter:    9   residual = 3.692025e-01
    iter:   10   residual = 1.010501e-01
    iter:   11   residual = 3.528043e-02
    iter:   12   residual = 6.867475e-03
    iter:   13   residual = 1.206750e-03
    iter:   14   residual = 9.768426e-05
    iter:   15   residual = 1.063625e-05
    iter:   16   residual = 1.902986e-06
    iter:   17   residual = 3.433117e-07
    iter:   18   residual = 9.218341e-08

    Real residual is  1.48685e-09
Computing final fluxes

Total iterations:    18
Total elapsed time:  0.00 (sec.)
  GMRES setup time:  0.00 (sec.)
```

(continues on next page)

(continued from previous page)

```
  GMRES solve time:  0.00 (sec.)
Residual L2 norm:    1.48685e-09
Flux Integration:    -0.219045
Minimum Emissivity:  0.3
Maximum Emissivity:  1
Minimum Temperature: 443.752, Element: 778
Maximum Temperature: 1200, Element: 415
```

This output is somewhat similar to the Aria run-time output, in that the linear iteration count is given along with the linear residual, though the Chaparral output is more detailed, as it gives the linear residual at each linear step instead of a summary. Minimum and maximum values of emissivity and temperature in the enclosure can also help in understanding the behavior of the radiosity solve through time.

6.4 Diagnostic Output

Aria is instrumented with a diagnostic output capability that can be very useful when you're debugging problems. You can tell Aria to write additional information to the log file by enabling different "print masks". A list of common diagnostic options is summarized in the list below. For the complete list you can refer to the Aria runtime help which is printed if you add `-h` or `--help` to your launch command (e.g. `aria --help`).

- `--tftklog expression` - This option prints out the full expression tree to the log file. It can be useful for diagnosing missing expression errors.
- `--arialog nonlinear` - Output detailed nonlinear solve information in the log file and the exodus output file at each nonlinear iteration.
- `--tftklog finite_check` - Do additional checks for NaN or Inf values from expressions and output additional diagnostics.
- `--tftklog sens_check` - Perform a numerical differentiation of all expressions to check that their provided sensitivities are correct.
- `--arialog bc` - Display additional boundary condition information in the log file.

Another way to enable a print mask is more versatile and is done through your aria input file. At the highest level of the input file - somewhere in the `BEGIN SIERRA` block - you can add a `DIAGNOSTIC CONTROL` block. This enables you to specify intervals in time or time step number over which print masks should be enabled. As a simple example, let's say you want to enable the sensitivity checker (`--tftklog sens_check`) around time step number 47. Here's an example input file snippet that allows you to do that:

```
Begin Diagnostic Control tftklog
  From Step 46 to 48 enable sens_check
End
```

With this, the sensitivity checker will only be active for those three time steps (it's also valid to say `From Step 47 to 47 ...`). The *Diagnostic Control* section includes a detailed command reference for that input.

6.5 Troubleshooting

Sometimes an Aria job will fail. This can be for a number of reasons, including invalid syntax in the input file, an ill-posed or non-physical problem setup, insufficient mesh resolution, computing hardware issues, or other reasons. When this happens, you will need to look for clues in the log file, the stdout stream (or slurm file), and any results output files (Exodus or Heartbeat). The following sections address some of the common failure modes you might encounter.

Invalid Input Syntax

If you use something in the input file that is not valid Aria syntax, you will get an error in the log file. If the problem is invalid syntax, the error will be at the point in the input file where the invalid syntax was, and will suggest alternative syntax. For example, if we use the wrong syntax for a constant thermal conductivity (using `tk` instead of `value` or `k` for the constant)

```
BEGIN ARIA MATERIAL Kryptonite
  Thermal Conductivity = constant tk=1.0
```

The first signal we get that something is wrong is in the stdout stream (or the slurm file if launched on an HPC), but it does not tell us what the specific error is.

```
$ aria -i demo.i
exception on all processors:
Execution terminated due to errors

*** SIERRA ABORT on P0 ***
*** check the log file for more information ***
```

Looking in the log file, at the end of the file we would see

```
There was 1 error encountered during parse
There were no warnings encountered during parse
Execution terminated due to errors
```

(continues on next page)

(continued from previous page)

```
SIERRA execution failed during parse with the following exception:
Execution terminated due to errors
```

This does not tell us what the error is by itself either, but since it says it was a parse error we should search for the keyword "error" in the section of the log file where it repeats the input file. Then we find

```
  BEGIN SIERRA myJob
    BEGIN ARIA MATERIAL Kryptonite
      Thermal Conductivity = constant tk=1.0
demo.i:7: Error: No matching command line found for 'Thermal Conductivity␣
↪= constant tk=1.0'
demo.i:7: The following command lines with matching keywords are defined:
demo.i:7:   Thermal Conductivity [ {Of|Species|Subindex} <speciesname:␣
↪string> ] = Constant {K|Value} = <k: real>
demo.i:7:
      heat conduction     = basic
```

The error message tells us which line was invalid, and shows the syntax for the best match. In this case, the model we used (`constant`) is a valid model, so it only prints the syntax for that model. If we had used an invalid model (e.g. `constnt`), the list of possibilities would include all the valid models for thermal conductivity (shown below with a truncated list)

```
  BEGIN ARIA MATERIAL Kryptonite
    Thermal Conductivity = constnt k=1.0
demo.i:7: Error: No matching command line found for 'Thermal Conductivity␣
↪= constnt k=1.0'
demo.i:7: The following command lines with matching keywords are defined:
demo.i:7:   Thermal Conductivity [ {Of|Species|Subindex} <speciesname:␣
↪string> ] = T_Exponent K_Ref = <k_ref:
demo.i:7:      real>  T_Ref = <t_ref: real>  N = <n: real>
demo.i:7:   Thermal Conductivity [ {Of|Species|Subindex} <speciesname:␣
↪string> ] = Volume_Average
demo.i:7:   Thermal Conductivity [ {Of|Species|Subindex} <speciesname:␣
↪string> ] = Summed Contributions =
demo.i:7:      <contributions: string>
...
```

Deprecated Syntax

Most Aria problems use stable syntax that does not change from one release to the next, however sometimes features need to be removed for maintainability and sustainability of the codebase. This is done using a deprecation cycle to allow for time to migrate input file syntax.

Aria uses a 6 to 12 month deprecation cycle (or longer for larger feature deprecations). When a deprecation is introduced, it will result in a warning immediately, which shows up both in the log file and in the stderr/slurm stream. For example, if you used a deprecated post-processor in Aria version 5.10 you would see the following warning in the stderr/slurm output

```
WARNING: Deprecated feature removed in Version 5.11 detected.
The 'POST PROCESS FLUX' solution options command is deprecated in favor␣
↪of the new postprocessing syntax. Replace
  BEGIN SOLUTION OPTIONS
    post process FLUX heat_conduction on fluid_interface as HeatFlux
  END SOLUTION OPTIONS
with
  postprocess expression_flux of expression heat_conduction on fluid_
↪interface as HeatFlux
for a direct replacement. See Sections 21.5 of the user manual for more␣
↪information.
Please email sierra-help@sandia.gov if you have issues converting post␣
↪process commands.
```

The same warning would also show up in the log file

```
demo.i:82: Warning: Deprecated feature removed in Version 5.11 detected.
demo.i:82: The 'POST PROCESS FLUX' solution options command is deprecated␣
↪in favor of the new postprocessing
demo.i:82:   syntax. Replace
demo.i:82:   BEGIN SOLUTION OPTIONS
demo.i:82:     post process FLUX heat_conduction on fluid_interface as␣
↪HeatFlux
demo.i:82:   END SOLUTION OPTIONS
demo.i:82: with
demo.i:82:   postprocess expression_flux of expression heat_conduction on␣
↪fluid_interface as HeatFlux
demo.i:82: for a direct replacement. See Sections 21.5 of the user manual␣
↪for more information.
demo.i:82: Please email sierra-help@sandia.gov if you have issues␣
↪converting post process commands.
demo.i:82: in line command POST PROCESS FLUX
```

Deprecation version numbers are always odd-numbered (non-release). When the 5.10 version is

released, the version of `sierra/daily` is set to 5.11. In this example, this means that using the deprecated post-processor will be an error if you run `sierra/daily` during the 5.11 release cycle, and in the next release (5.12) it will be an error. Pay attention to these warnings and update your inputs in a timely manner to avoid getting errors at future releases.

Non-Physical Temperatures

The conservation of energy equation in its continuous form (and in the absence of source terms) satisfies a maximum principle. That is, for a steady-state problem the maximum and minimum temperatures in the domain must occur on the boundary of the domain, and for a transient problem on either the boundary or the initial condition. However, the discretized form of the energy equation does not necessarily satisfy a discrete version of the maximum principle (DMP). In particular for the standard Galerkin finite element method used in Aria there are conditions on the mesh quality that must be met for the diffusion operator to satisfy a DMP.

The article "Don't suppress the wiggles—They're telling you something!" can provide useful context on the nature of oscillations in finite element solutions and provides valuable background information as to why they occur and what they mean for your solution.

In general, if you encounter non-physical high or low temperatures the troubleshooting steps you should start with are:

1. Examine the mesh in the region of spurious temperature to ensure you have enough resolution in the direction of heat flow. Pay particular attention to thin regions and make sure they are more than 1 element thick. Also note that there can be issues on edge/corner elements that have multiple different boundary conditions applied to them, particularly with tetrahedral elements.
2. Try using first order time integration, lumped_mass, and lumped_flux boundary conditions (although this will increase discretization error)
3. Enable the flux limiter option if using a tetrahedral mesh (described below).

For linear tetrahedral elements the DMP condition is that there are no obtuse dihedral angles between faces of elements in the mesh [47]. Anisotropic thermal conductivity can also contribute to this effect [48]. Additionally, the consistent mass term can also lead to DMP violations in transient problems [49]. In practice this can lead to non-physical temperatures appearing in the solutions generated by Aria, most commonly when large heat fluxes are applied to an initially cold domain with a low thermal conductivity.

If non-physical temperature solutions are observed Aria supports an option to apply a nonlinear flux correction based on the work of Kuzmin et al. that restores a DMP on arbitrary meshes with arbitrary material properties [49, 50]. This option may be activated by using the `APPLY FLUX LIMITER STABILIZATION` line command. For transient problems it is also essential to use the `LUMPED_MASS` form of the time derivative term. At present, this option is only intended for use with linear elements, not second order 10 node tetrahedral elements or 27 node hexahedral

elements. Source terms also do not have any limiters applied to them at present, as a result they may still cause non-physical temperatures in some cases.

Additionally, applying the stabilization has several downsides so that we only recommend enabling it if problematic non-physical temperatures are observed without it. Solutions with the stabilization activated are more diffusive and have higher error than unstabilized solutions (though they do converge at the same order with mesh refinement). The stabilization operator is also nonlinear and can adversely affect the convergence of the Newton-Raphson iteration within each time step. Combined with the cost of calculating the stabilization terms this can have noticeable impact on simulation runtime.

Repeated Time Step Failure

If your problem *fails a timestep*, Aria will decrease the step size and attempt the solve(s) again. If this keeps happening, eventually Aria will fail when the time step reaches the allowable minimum. Some strategies to try if you encounter this type of failure are:

1. Ensure that your problem is mathematically well-posed and that the linear solves are completing with a sufficiently low final residual (they should be lower than your nonlinear residual target).
2. Increase the number of nonlinear iterations if the problem was converging but not reaching your target residual before hitting the iteration limit.
3. Change the nonlinear solution strategy from `Newton` to `Line_search` to enable an adaptive under-relaxation that attempts to ensure that each nonlinear step always results in a reduction in the nonlinear residual. This will increase the cost of each nonlinear iteration.

Linear Solver Failure

If the linear solver fails to solve your linear system there will be an error code reported in the log file. The *Solver Error Codes* section outlines the meaning behind these error codes. In general, if you see error codes you should re-examine your choice of linear solver or check whether the problem as-posed is non-singular. The *Solver Selection Guidelines* section has suggestions for how to incrementally adjust the solver settings for harder-to-solve problems.

Out of Memory

When running memory intensive computations like calculating view factors on memory limited HPC platforms, your simulation will be killed if the compute node runs out of memory. This can also happen if you run a conventional problem with too many elements per core. Unfortunately there is no hard guideline as to the maximum problem size since it depends on what options are used in the simulation and varies between HPC platforms, but without enclosure radiation it should be able to support several million elements per core before running into memory issues.

When a simulation on the HPC runs out of memory, the user who launched it will typically receive an automated email from the HPC system notifying them that their job ran out of memory, although this notification may not be sent immediately. If your jobs are running out of memory you generally have a few options:

1. Run the job on a larger number of nodes or cores
2. Use fewer cores per node than the total on an HPC platform (e.g. request twice as many nodes but use half as many cores per node)
3. Run the job on a different system that has more memory per core available

Since most memory-limited simulations in Aria are due to view factor calculations for enclosure radiation, the Chapparal log file also includes estimates on memory use to help you size your job submission appropriately.

6.6 Guidelines

The present section offers some useful guidelines for various user settings, which can be a useful starting point in setting up an analysis.

Initial Time Step Estimation for Thermal Problems

A method for estimating an appropriate initial time step is outlined below. This procedure is originally due to Levi ([51]) though it has been discussed by several other authors ([52, 53, 54]).

In the following development, it is assumed that a finite element mesh has been constructed that will adequately model the thermal phenomena of interest (e.g., thermal shock problems will require a fine mesh near a boundary while slower thermal transient will be less demanding on mesh refinement). For a given spatial discretization, a local characteristic length, Δx, is chosen based on element size. Typically, this characteristic length is measured normal to a boundary on which a temperature or heat flux (source) disturbance occurs. Based on the characteristic length, the local heat transfer coefficient, h, and the local thermal conductivity, k, a local element Biot number ($Bi = h\Delta x/k$) can be computed. Note that for a prescribed temperature, heat flux or heat source boundary condition, the heat transfer coefficient, h, is assumed to be large leading to a large Biot number.

In order to bound the thermal gradient that will occur at the boundary in the first time step, the ratio of the temperature at a distance Δx (characteristic length) from the boundary to the temperature on the boundary is selected. Let this ratio be defined by $\Theta = T(\Delta x)/T_{surface}$. Typical values of Θ will range from 0.10 to 0.25. With the estimated values for local Biot number and temperature ratio Θ, a local Fourier number ($Fo = \alpha\Delta t/\Delta x^2$) may be found from the charts in Fig. 6.1 and Fig. 6.1. These graphs are based on an analytic solution for one-dimensional conduction with a convective boundary condition ([55]). A value of the local Fourier number and values for the characteristic length and thermal diffusivity, α, then allow a time step to by computed.

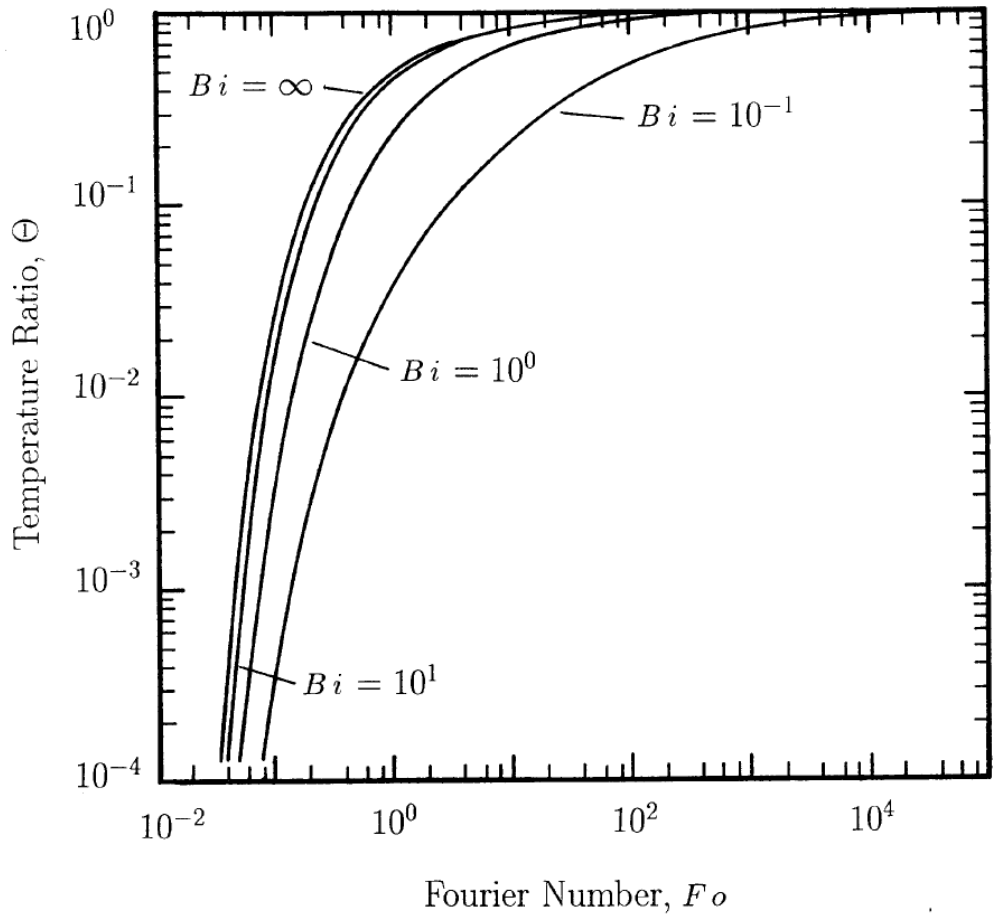


Fig. 6.1: Temperature ratio versus Fourier number for various Biot numbers

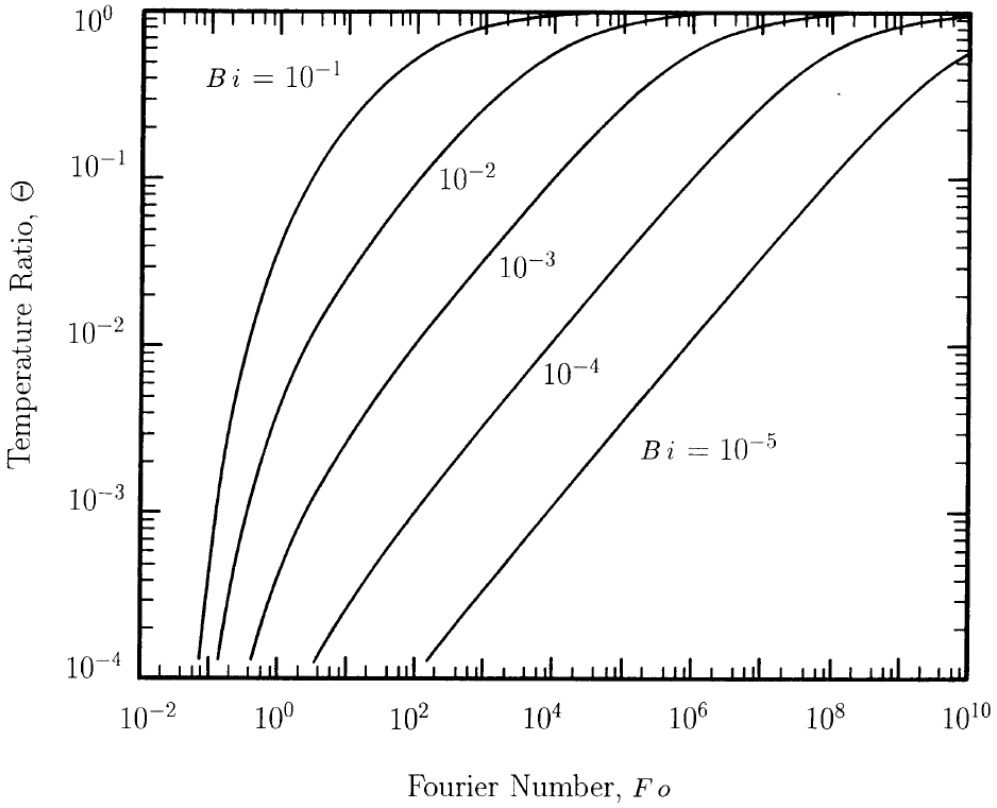
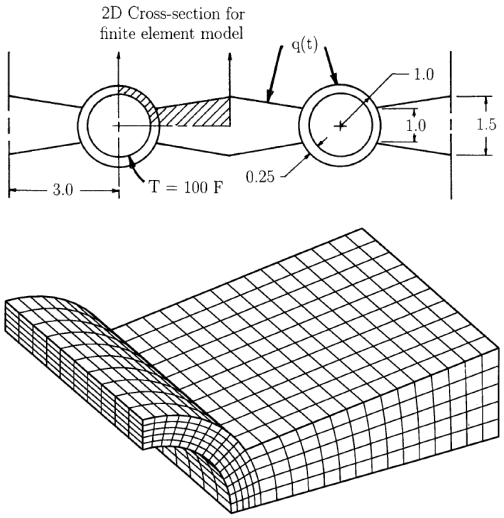


Fig. 6.2: Temperature ratio versus Fourier number for various Biot numbers

As an example of the procedure outlined above consider the transient problem described in the following example:



Geometry and mesh of finned radiator problem (Dimensions are in inches)

Consider the transient thermal response of a finned tube radiator, as shown in *Finned Radiator*. Thermal properties for the radiator are as follows:

$$\begin{aligned}\rho &= 2.7071 \times 10^3 \text{kg/m}^3 \text{‘} \\ C &= 870.854 \text{J/kg} \cdot \text{K} \\ k &= 207.566 \text{W/m} \cdot \text{K} \\ \alpha &= 8.806 \times 10^{-5} \text{m}^2/\text{s}\end{aligned}$$

The characteristic size is based in the average size of the elements along the radiator fin, as shown in *Finned Radiator*, and is equal to $\Delta x = 3.175 \times 10^{-3}$ m. The boundary condition is an applied heat flux, so the Biot number is assumed large (infinite). Using a temperature ratio of $\Theta = 0.10$, then Fig. 6.1 yields a Fourier number of $Fo \approx 0.20$. Thus,

$$\Delta t = \frac{(Fo)(\Delta x^2)}{(\alpha)} = \frac{(0.20)(3.175 \times 10^{-3})^2}{(8.806 \times 10^{-5})} = 0.023 \text{ seconds.}$$

The above procedure can also be used to estimate the overall response time for a region. In this case the length scale is a characteristic length for the entire region and the temperature ratio should be order unity. The Fourier number will then produce the approximate time interval required to reach equilibrium.

6.7 FAQ

Applying complex BCs

The best way to do this is to define a surface material. A surface material lets you use *user expressions*, *string functions*, or any of the other generic expression types to define things like convection coefficient or reference temperature. An example showing how to include a complex convection boundary condition is shown below.

```
# Define the material
begin aria material cht_material
    heat transfer coefficient = scalar_string_function f = "10 + 5*x"
    user expression = user_function user_tag=Tref_scaling name=Tref_table␣
 ↪X=time
    bc reference temperature = scalar_string_function f = "Tref_scaling +␣
 ↪10*x"
end
```

```
# Apply the material to the relevant sideset(s)
Begin finite element model mymesh
    #...
    Use material cht_material for surface_1
End
```

```
# Apply the generalized nat conv BC on the relevant sideset(s), this
# will apply Flux = h*(T - Tref) using the h and Tref you defined in your␣
 ↪material
BC Flux for Energy on surface_1 = Generalized_Nat_Conv
```

If your convective BC only uses string functions for h and T_{ref}, they can also be supplied in-line instead of with a surface material.

```
BC Flux for Energy on surface_1 = Generalized_Nat_Conv \$
    h = "10 + 5*x"                                     \$
    T_ref = "300 + 50*sin(t) + 10*x"
```

Missing expression error

When setting up your problem, you may encounter a missing expression error when you run Aria. This will typically start with:

```
Missing expression FOO on subdomain [Etet_Te4_g4/Etet_Te4_g4/Etet_Te4_g4␣
↪on block_3/Etet_Te4_g4_default_integration_rule/-1]
```

An *expression* in Aria is another name for a defined property in the *expression graph* (like `DENSITY` or `SPECIFIC_HEAT`). If you see an error about a missing expression `FOO`, it means that some model you are using has listed `FOO` as a prerequisite, but you have not defined a model for `FOO`. The error message will also tell you which mesh block `FOO` is missing on (`block_3` in the example above) - so the solution is usually to add a definition for `FOO` in the material used on that block.

For example, if you had defined `DENSITY` using some model that uses `FOO` as an input but did not define `FOO` you would receive this error.

```
Begin Aria material blah
    # this fictitous model calculates density = 10 + foo
    Density = Computed_From_Foo

    # If we don't define "foo" we will get a missing expression error
    # Foo = constant value = 5
End
```

If this error occurs on a block subdomain tag, the property is likely missing in the associated block's *material property definition*. For surface subdomains (e.g. surface BCs, interface BCs, etc) the expression can be defined either in the surface material (if defined) or the owning block's material. For contact enforcements, properties can still be defined in either the surface of block materials, but can also be provided as *contact material properties*.

Expression Flux Limitations

Explanation

An expression flux postprocessor is often used for postprocessing on internal surfaces. Depending on the resolution and geometry a problem, one might find that the results from these postprocessors inaccurate. The root issue comes from the fact that Aria uses a finite element approximation. As opposed to a control-volume approach, a finite element approach does not generally guarantee equality of flux along element boundary. Within these elements, the gradient is recovered only up to a given order. Since expression flux postprocessors use the element's gradient operator they are subject to these discretization errors.

To demonstrate this, consider the following example of two Quad4 elements sharing a face. For the linear function x, the elements can exactly recover the field solution and its constant gradient.

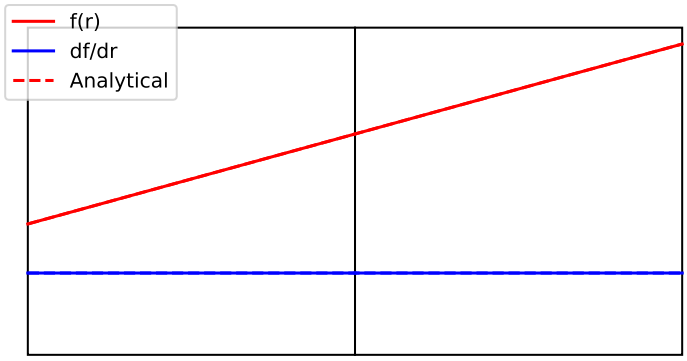


Fig. 6.3: Interpolated function and gradient for linear function

For the cubic function x^3 on the other hand, the recovered gradient is not exact

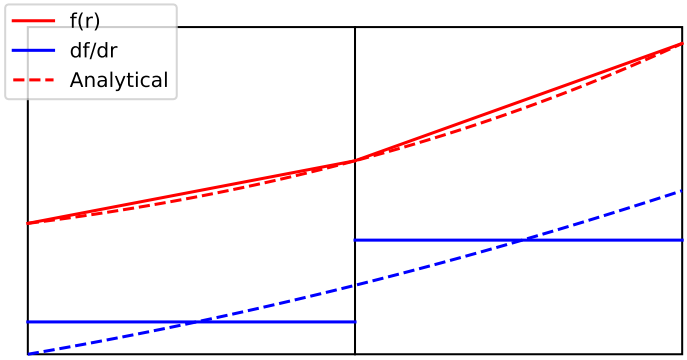


Fig. 6.4: Interpolated function and gradient for cubic function

As an example of the implication of this, consider the following domain

In this study, the elements are shrunk to maintain the same aspect ratios with increasing resolution along the circumferential direction. The following quantities are calculated over the shared internal surface Γ at each resolution and included in the convergence plot below

Circumference

$\int_\Gamma d\Gamma$

Radius

$\frac{\int_\Gamma r d\Gamma}{\int_\Gamma d\Gamma}$

Flux

$\int_\Gamma \nabla f \cdot \hat{n}\, d\Gamma$ where the gradient of $f = r^2 + (r_{int}\theta)^2$ is evaluated using the

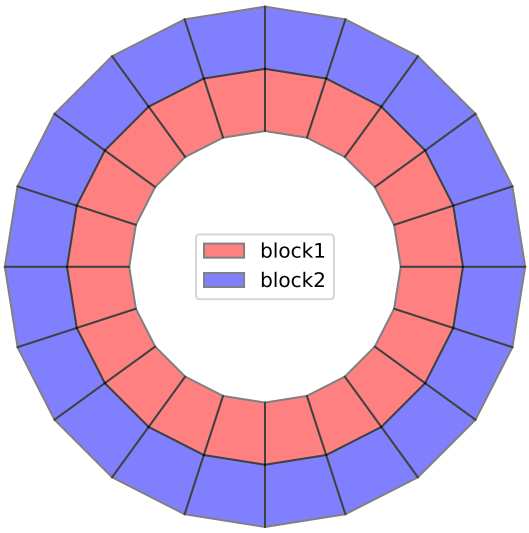


Fig. 6.5: Convergence study domain

element's gradient operator

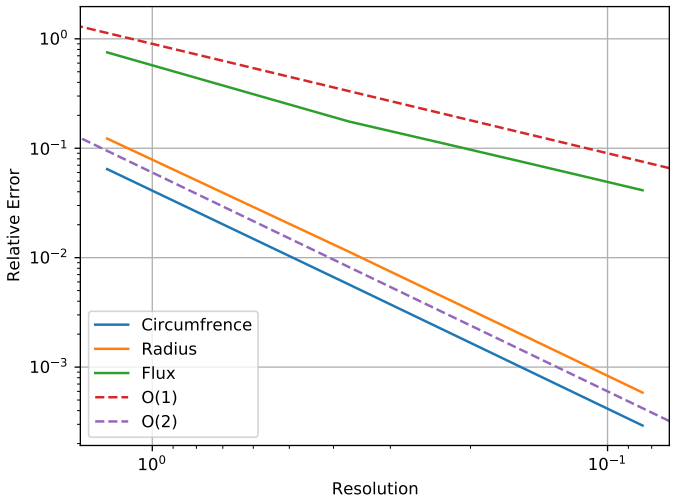


Fig. 6.6: Convergence study

As is expected, both the radius and circumference converge with O(2) with the linear elements used. Even for this simple geometry, though, the initial error of the flux term is a decade higher than that of the solution field. The convergence is also O(1) for the gradient-based term, as opposed to the solution field operations converging with O(2). This demonstration highlights why one might need to refine much more than is necessary to attain the proper finite element solution when using shape function gradients for postprocessing.

Solutions

The following section lists the possible solutions that can be taken, listed in order of their ease of implementing.

Not on Internal surface? Replace with residual flux postprocessor.

If it possible to isolate the flux component of a term in a residual-flux postprocessor, this will always be the easiest way. For example for some `external_surface`, one should postprocess the flux term separately and feed that into the remainder of the calculation e.g.

```
# instead of using HEAT_CONDUCTION expression
postprocess integrated_flux of vector_string_function \$
  f_x = "HEAT_CONDUCTION_X * density" \$
  f_y = "HEAT_CONDUCTION_Y * density" \$
  f_z = "HEAT_CONDUCTION_Z * density" \$
  on external_surface as foo_w_expr

#postprocess the flux then feed it into an expression
postprocess FLUX_SCALAR_FIELD of equation energy on external_surface␣
 ↪as energy_flux
postprocess integral of function "energy_flux * density" on external_
 ↪surface as foo_w_residual_flux
```

Average flux over both touching extents.

For some `internal_surface` touching both blocks 1 and 2, performing the average over both touching extents will achieve a closer approximation since it averages the gradient contributions of both elements. As was demonstrated above, the average between the two elements will approach the true value at the face more quickly than each element's values. Note that one of the terms must have a sign flipped so that the opposing normal directions do not cancel out.

```
postprocess integrated_flux of expression HEAT_CONDUCTION on␣
 ↪internal_surface touching block_1 as heat_flux_b1
postprocess integrated_flux of expression HEAT_CONDUCTION on␣
 ↪internal_surface touching block_2 as heat_flux_b2
postprocess global_function "(heat_flux_b1 - heat_flux_b2)/2" as␣
 ↪heat_flux
```

Refine mesh either globally or locally.

If this term still is found to be too far off, refining the mesh can help. One option here is to locally refine the mesh along the surface of interest in the mesh generation software. If this portion of the mesh has a high gradient, *adaptivity* can be enabled in order to locally refine the interface using e.g.

```
# Scope: Sierra
Begin Recovery Indicator recover_ind
  Store In indicator_field
  Use Function solution->temperature
  Recover gradient
End
Begin Percent Of Max Marker max_mark
  Store In marker_field
  Use Element Field indicator_field
  Percent Of Max To Refine Above 80
  Percent Of Max To Coarsen Below 10
  Maximum Refinement Level 5
End
...
Begin Transient time_stepping_block
  advance AriaRegion

  IndicateMarkAdapt AriaRegion Using recover_ind max_mark \$
    when "transient_time_stepping_block.CURRENT_STEP % 5 == 0"
End Transient
```

Extract net flux from volume integrated quantities.

Finally if the mesh cannot be refined, one could keep track of the net generation of the equation and subtract off any sources to get the net flux out of the body. Since this approach avoids any gradient-based quantities, it will give a notably more accurate result.

```
postprocess integral of function "dt_temperature*specific_
 ↪heat*density" on each_block as net_energy_gen
postprocess integral of expression energy_source on each_block as␣
 ↪internal_energy_src

# Calculate the component of energy generation that came from fluxes
postprocess global_function "net_energy_gen_block_1 - internal_
 ↪energy_src_block_1" as net_energy_flux_block_1
postprocess global_function "net_energy_gen_block_2 - internal_
 ↪energy_src_block_2" as net_energy_flux_block_2
postprocess global_function "net_energy_gen_block_3 - internal_
 ↪energy_src_block_3" as net_energy_flux_block_3
```

Subtract the component due to external flux via residual-flux postprocessor.

If one is interested in the relative contributions of internal and external fluxes, a residual flux postprocessor can be run over the external surfaces of each block. Again since these postprocessors use the residual of the true equations solved, they will not be subject to the discretization errors discussed above

```
# Find the external component of the fluxes on each body
postprocess integrated_flux of equation energy on b1_ext_surf as net_
 ↪ext_flux_block_1
postprocess integrated_flux of equation energy on b2_ext_surf as net_
 ↪ext_flux_block_2
postprocess integrated_flux of equation energy on b3_ext_surf as net_
 ↪ext_flux_block_3

# What remains is the net internal fluxes
postprocess global_function "net_energy_flux_block_1 - net_ext_flux_
 ↪block_1" as net_int_flux_block_1
postprocess global_function "net_energy_flux_block_2 - net_ext_flux_
 ↪block_2" as net_int_flux_block_2
postprocess global_function "net_energy_flux_block_3 - net_ext_flux_
 ↪block_3" as net_int_flux_block_3
```

Postprocess relative internal flux across blocks

Finally, if one wants to know the relative internal flux between each block, the `net_int_flux` can be written to a a heartbeat file. The relative internal fluxes can then be back-calculated in postprocessing of the results e.g.

```
import numpy as np

# 1->2, 2->3, 3->1
connectivity = np.array([
  [ 1, 0,-1], #block 1
  [-1, 1, 0], #block 2
  [ 0,-1, 1], #block 3
])

net_int_flux = np.array([
  net_int_flux_block_1,
  net_int_flux_block_2,
  net_int_flux_block_3,
])

# Overwrite row and fix mean internal flux at zero
net_int_flux[-1] = 0
connectivity[-1,:] = 1

fluxComps = np.linalg.solve(connectivity, net_int_flux)

rel_internal_flux_1to2 = fluxComps[0]
```

(continues on next page)

(continued from previous page)

```
rel_internal_flux_2to3 = fluxComps[1]
rel_internal_flux_3to1 = fluxComps[2]
```

Note that the original linear system is singular (any solution +C satisfies the net flux condition), we must override one of the rows and the solution is only the *RELATIVE* flux between the blocks.

Results output failure for large meshes

For very large meshes, the numerical global IDs for any entities in the mesh (nodes, elements, etc.) may become very large; so large that they exceed the 32-bit limit. If this is the case, make sure the results block uses the 64-bit limit, which is enabled with the following command in the results block:

```
Enable Large IDs
```

This defaults to using a netCDF-4 variant output, which enables 64-bit IDs. Other options for modifying results output, or possibly adding extra flags for debugging in the results block can be found in *Results Output*.

There have been reported cases of output failure occurring, even with netCDF-4 output variants enabled. This will typically result in an obscure error in the log file, but the output error (the slurm file of a HPC run) will have an output that is similar to the below:

```
Exodus Library Warning/Error: [~Redefine] in file '<file_output_name>'
    Error: failed to complete variable definitions in file id 131072
    NetCDF: HDF error
```

This error is internal to the HDF5/netCDF library, and related to the netCDF-4 variant itself. An immediate fix that has worked for some cases is to use a netCDF-5 variant output instead. This can be done by setting the following command in the results output property block:

```
property FILE_TYPE = netcdf5
```

This command is available to all users, but not listed in the command reference in *Results Output* as this capability is still being developed. If this does not fix the issue, please forward this issue to the Sierra help desk.

7 Command References

This section is a comprehensive list of command summaries recognized by the Sierra parser. In general, these summaries include the **Syntax** of how to specify the command, the **Scope** where it can be used, and a **Summary / Description** of what effect it has. User inputs are stylized as follows for convenience:

Representation	Description
string	A string (possibly quoted) argument
number	A number (float or integer) input
{enum}	An enumeration of possible options (listed in parameter table). They are enclosed in { } brackets.
[Optional]	Arguments that augment the line but not required. They are enclosed in *[]* brackets.
(Cmd = X | [Other = Y])	A group of commands that can be in any order. May include optional and required commands. They are enclosed in *()* brackets and separated by |. Optional commands within the list are enclosed in *[]*.

The command summaries are included by topic below:

Sierra Commands

7.1 Adaptivity Settings

Adaptivity Options

Scope

Sierra

Summary

Used to define markers and indicators used for adaptivity

```
begin Adaptivity Options AdaptBlockName

  begin Function Marker MarkerName
  end

  begin Uniform Marker MarkerName
  end

  begin Field Indicator IndicatorName
```

```
   end

   begin Gradient Error Indicator IndicatorName
   end

   begin Signed Distance Indicator IndicatorName
   end

end Adaptivity Options AdaptBlockName
```

Adaptiveloop

Scope

Sequential, Solution Control Description, System, Transient

Summary

Specify block-style post processing command

```
begin Adaptiveloop Name

   Advance Name... [ When When-expression  ]

   Adapt Mesh For AdaptRegionName Using AdaptBlockName [When␣
↪WhenExpression]

   Event Name... [ When When-expression  ]

   Postprocess Aria Region RegionName ([Equation System␣
↪EquationSystemName]  | [When WhenExpression])

   Transfer Name [ When When-expression  ]

   begin Sequential Name
   end

   begin Transient Name
   end

end Adaptiveloop Name
```

Line Commands

Advance

Syntax

Advance *Name. . .* [When *When-expression*]

Summary

Used within a Solver Control block to indicate a single step that advances the solution. The name is that matches the physics.

Parameter	Value	Default
Name	string. . .	–

Adapt Mesh

Syntax

Adapt Mesh For *AdaptRegionName* Using *AdaptBlockName* [When *WhenExpression*]

Summary

Adapt the mesh using the adaptive command block name

Parameter	Value	Default
AdaptRegionName	string	–
AdaptBlockName	string	–
WhenExpression	(expression)	–

Event

Syntax

Event *Name. . .* [When *When-expression*]

Summary

Used within a Solver Control block to indicate a single step that has no time associated with it. It can cause a solution transfer between regions or cause something to print.

Parameter	Value	Default
Name	string. . .	–

Postprocess Aria Region

Syntax

Postprocess Aria Region *RegionName* ([Equation System *EquationSystemName*] | [When *WhenExpression*])

Summary

Run Aria Region postprocessors. If specified, only the single equation system's PPs are run.

Parameter	Value	Default
RegionName	string	–
EquationSystemName	string	–
WhenExpression	(expression)	–

Transfer

Syntax

Transfer *Name* [When *When-expression*]

Summary

A Solver Control Transfer line command which executes all transfers defined from the specified region. All transfers with a send region of 'name' will be executed.

Parameter	Value	Default
Name	string	–

Field Indicator

Scope

Adaptivity Options

Summary

An indicator calculated from a nodal field

Description

This calculates an element field with the element centroid value defined by the specified nodal field. Global variables for the min, max, and average of the field are also calculated.

```
begin Field Indicator IndicatorName

   Location = Value...

   Nodal Field = Value

   Reduction = {centroid | max | min}

end Field Indicator IndicatorName
```

Line Commands

Location

Syntax

Location = *Value. . .*

Summary

A block or list of blocks defining the location of indicator

Parameter	Value	Default
Value	string. . .	–

Nodal Field

Syntax

Nodal Field = *Value*

Summary

Nodal field name to do the element centroid projection on

Parameter	Value	Default
Value	string	–

Reduction

Syntax

Reduction = *{centroid | max | min}*

Summary

Project nodal field onto element centroid, max of nodes on element, or min of nodes on element. DEFAULT = CENTROID

Parameter	Value	Default
reduction_enumeration	{centroid | max | min}	–

Signed Distance Indicator

Scope

Adaptivity Options

Summary

An indicator based on distance from a nodal signed distance field

Description

This calculates the distance from an element to an interface defined by a nodal

signed distance field. If the element spans the interface its distance is 0. For other elements, its distance is the minimum distance of any of its nodes

```
begin Signed Distance Indicator IndicatorName

   Location = Value...

   Signed Distance Field = Value

end Signed Distance Indicator IndicatorName
```

Line Commands

Location

Syntax

Location = *Value...*

Summary

A block or list of blocks defining the location of indicator

Parameter	Value	Default
Value	string...	–

Signed Distance Field

Syntax

Signed Distance Field = *Value*

Summary

Nodal distance field name to calculate distance from

Parameter	Value	Default
Value	string	–

Gradient Error Indicator

Scope

Adaptivity Options

Summary

An indicator based on nonlinearity in the gradient of a nodal field

Description

This calculates a gradient-based error indicator using the difference between the

element-based gradient and a projected nodal gradient. The error indicator on a given element is the average of the error at each of its nodes, normalized by an element length scale.

The element gradient is the average of the gradient at each integration point:

$$\nabla\phi_e = \frac{1}{n_{ip}} \sum_{ip=1}^{n_{ip}} \nabla\phi_i p$$

Then the nodal gradient is assembled as a volume-weighted average of the gradient of the connected elements

$$\nabla\phi_n = \frac{\sum_{i=1}^{n_{elems}} \nabla\phi_{e,i} V_{e,i}}{\sum_{i=1}^{n_{elems}} V_{e,i}}$$

Finally, the gradient error per element is calculated as the average difference between the element gradient and its nodal gradients, scaled by a characteristic element length scale

$$E_g = \frac{V_e^{1/n_{dim}}}{n_{nodes}} \sum_{i=1}^{n_{nodes}} ||\nabla\phi_{n,i} - \nabla\phi_e||$$

Global variables for the min, max, and average of the final indicator error field are also calculated.

```
begin Gradient Error Indicator IndicatorName

   Location = Value...

   Nodal Field = Value

end Gradient Error Indicator IndicatorName
```

Line Commands

Location

Syntax

Location = *Value...*

Summary

A block or list of blocks defining the location of indicator

Parameter	Value	Default
Value	string...	–

Nodal Field

Syntax

Nodal Field = *Value*

Summary

Nodal field name to compute the gradient based error indicator

Parameter	Value	Default
Value	string	–

Uniform Marker

Scope

Adaptivity Options

Summary

Uniformly mark all blocks for refinement

Description

This will mark all elements in the specified block(s) for refinement until they reach their max refinement levels.

```
begin Uniform Marker MarkerName

   Location = Value...

   Maximum Refinement Level = MaxRefineLevel

end Uniform Marker MarkerName
```

Line Commands

Location

Syntax

Location = *Value...*

Summary

List block(s) or sideset(s) where marker should be applied. When a sideset marker is applied, its attached element is marked.

Parameter	Value	Default
Value	string...	–

Maximum Refinement Level

Syntax

Maximum Refinement Level = *MaxRefineLevel*

Summary

Set maximum refinement level for this marker

Parameter	Value	Default
MaxRefineLevel	integer	–

Function Marker

Scope

Adaptivity Options

Summary

Use string functions to define refinement and unrefinement rules

Description

This takes a user-defined string function for refinement and/or un-refinement. The function can include time (t), time step (dt), position (x, y, z), any global variable, or any element field. If both refinement and un-refinement rules are specified, the refinement rule will take precedence.

```
begin Function Marker MarkerName

   Location = Value...

   Maximum Refinement Level = MaxRefineLevel

   Refine When refinefunc

   Unrefine When unrefinefunc

end Function Marker MarkerName
```

Line Commands

Location

Syntax

Location = *Value...*

Summary

List block(s) or sideset(s) where marker should be applied. When a sideset marker is applied, its attached element is marked.

Parameter	Value	Default
Value	string. . .	–

Maximum Refinement Level

Syntax

Maximum Refinement Level = *MaxRefineLevel*

Summary

Set maximum refinement level for this marker

Parameter	Value	Default
MaxRefineLevel	integer	–

Refine

Syntax

Refine When *refinefunc*

Summary

String function condition for refinement based on indicators

Parameter	Value	Default
refinefunc	“string”	–

Unrefine

Syntax

Unrefine When *unrefinefunc*

Summary

String function condition for un-refinement based on indicators

Parameter	Value	Default
unrefinefunc	“string”	–

7.2 Diagnostic Control

Diagnostic Control

Scope

Sierra

Summary

Specifies the diagnostic writer to set output control features.

Description

The diagnostic and informational output can be selectively enabled based on time, step or an application specified condition. During the application's procedure execution loop, the diagnostic controller evaluates the enclosed line commands in the order specified in the input deck. The diagnostic options specified in the first line command that meets its criteria are applied.

Since control parameters are only applied when the criteria is met, it is important to include an ENABLE line command with the base settings to be applied as a baseline. Refer the '-h' output for a complete list of diagnostic writers and available values.

The options which may be enabled varies for each application, diagnostic and information writer. To obtain a list of available options, use the sierra app -i -O -h command.

See Diagnostic Stream for specifying the output destination.

```
begin Diagnostic Control Name

   Enable Printmask

   From Step StartStep To EndStep Enable Printmask

   From Time StartTime To EndTime Enable Printmask

   Set Information Stream Path File_path

end Diagnostic Control Name
```

Line Commands

Enable

Syntax

Enable *Printmask*

Summary

Specifies the options to enable when no other control option criteria are satisfied.

Parameter	Value	Default
Printmask	"string"	–

From Step

Syntax

From Step *StartStep* To *EndStep* Enable *Printmask*

Summary

Specifies the options to enable when the step is within the specified range.

Parameter	Value	Default
StartStep	integer	–
EndStep	integer	–
Printmask	"string"	–

From Time

Syntax

From Time *StartTime* To *EndTime* Enable *Printmask*

Summary

Specifies the options to enable when the time is within the specified range.

Parameter	Value	Default
StartTime	real	–
EndTime	real	–
Printmask	"string"	–

Set Information Stream Path

Syntax

Set Information Stream Path *File_path*

Summary

File path to information stream information to.

Parameter	Value	Default
File_path	string	–

Diagnostic Stream

Syntax

Diagnostic Stream *File_name* [*Indent-streambuf-flags1[Indent-streambuf-flags2]*]

Summary

File path to write diagnostic messages to.

Parameter	Value	Default
File_name	string	–

7.3 Finite Element Model Parameters

Finite Element Model

Scope

Sierra

Summary

Describes the location and type of the input stream used for defining a geometry model for the enclosing region.

```
begin Finite Element Model Finite-Element-Model-Name

  Alias DatabaseName As InternalName

  Component Separator Character {=} Separator

  Create {edgeset | elemset | faceset | nodeset | sideset | surface}␣
↪NewSurfaceName Add SurfaceName...

  Coordinate System {=} {axisymmetric | barycentric | cartesian |␣
↪cyclidic | cylindrical | polar | quadriplanar | skew | spherical |␣
↪toroidal | trilinear}

  Database Name {=} StreamName

  Database Type {=} {catalyst | catalyst_exodus | cgns | dof | dof_
↪exodus | exodus | exodusii | exonull | generated | genesis | null |␣
↪parallel_exodus | textmesh}

  Decomposition Method {=} {block | cyclic | external | geom_kway | hsfc␣
↪| kway | kway_geom | linear | map | metis_sfc | random | rcb | rib |␣
↪variable}

  Omit Assembly AssemblyList...

  Omit Block BlockList...

  Omit Volume VolumeList...
```

```
Time Scale Factor {=} Scale

Use Generic Names

Use Material MaterialName For VolumeList...

begin Assembly Assembly_Name
end

begin Block Blockname
end

begin Parameters For Block Blockname
end

begin Parameters For Phase Phase Name
end

begin Parameters For Surface Surface_Name
end
```

end Finite Element Model *Finite-Element-Model-Name*

Line Commands

Alias

Syntax

Alias *DatabaseName* As *InternalName*

Summary

Name the database entity "DatabaseName" as "InternalName"

Description

This "InternalName" may then be referenced in the data file in addition to the original name.

Parameter	Value	Default
DatabaseName	string	–
InternalName	string	–

Component Separator Character

Syntax

Component Separator Character *{=} Separator*

Summary

The separator is the single character used to separate the output variable basename (e.g. “stress”) from the suffices (e.g. “xx”, “yy”) when displaying the names of the individual variable components. For example, the default separator is “_”, which results in names similar to “stress_xx”, “stress_yy”, . . . “stress_zx”. To eliminate the separator, specify an empty string (“”) or NONE.

Parameter	Value	Default
{=}	{= | is}	–
Separator	string	–

Create

Syntax

Create *{edgeset | elemset | faceset | nodeset | sideset | surface}* *NewSurfaceName*
Add *SurfaceName. . .*

Summary

Create a new set (node, edge, face, element, side/surface) as the union of two or more existing sets. The sets must exist in the mesh database or have been created by a previous CREATE command.

Parameter	Value	Default
GroupType	{edgeset | elemset | faceset | nodeset | sideset | surface}	–
NewSurfaceName	string	–
SurfaceName	string. . .	–

Coordinate System

Syntax

Coordinate System *{=} {axisymmetric | barycentric | cartesian | cyclidic | cylindrical | polar | quadriplanar | skew | spherical | toroidal | trilinear}*

Summary

The interpretation of the geometry data stored in this database. Optional. Defaults to Cartesian.

Parameter	Value	De-fault
{=}	{= | are | is}	–
Coordi-nateSystem	{axisymmetric | barycentric | cartesian | cyclidic | cylindrical | polar | quadri-planar | skew | spherical | toroidal | trilinear}	–

Database Name

Syntax

Database Name *{=}* *StreamName*

Summary

The base name of the database containing the output results. If the filename begins with the '/' character, it is an absolute path; otherwise, the path to the current directory will be prepended to the name. If this line is omitted, then a filename will be created from the basename of the input file with a ".g" suffix appended.

Parameter	Value	Default
{=}	{= | are | is}	–
StreamName	string	–

Database Type

Syntax

Database Type *{=}* *{catalyst | catalyst_exodus | cgns | dof | dof_exodus | exodus | exodusii | exonull | generated | genesis | null | parallel_exodus | textmesh}*

Summary

The database type/format used for the mesh.

Parameter	Value	Default
{=}	{= | are | is}	–
Database-Types	{catalyst | catalyst_exodus | cgns | dof | dof_exodus | exodus | exodusii | exonull | generated | genesis | null | parallel_exodus | textmesh}	–

Decomposition Method

Syntax

Decomposition Method *{=}* *{block | cyclic | external | geom_kway | hsfc | kway | kway_geom | linear | map | metis_sfc | random | rcb | rib | variable}*

Summary

The decomposition algorithm to be used to partition elements to each processor in a parallel run.

Parameter	Value	Default
{=}	{= | are | is}	–
Method	{block | cyclic | external | geom_kway | hsfc | kway | kway_geom | linear | map | metis_sfc | random | rcb | rib | variable}	–

Omit Assembly

Syntax

Omit Assembly *AssemblyList. . .*

Summary

Specifies that the element blocks that are in the assemblies in AssemblyList will be omitted from the analysis.

Description

If an assembly is used to omit an element block, then it is illegal to refer to that element block later in the file. Any of the element blocks omitted will be removed from any assembly that contains them.

Parameter	Value	Default
AssemblyList	string. . .	–

Omit Block

Syntax

Omit Block *BlockList. . .*

Summary

Specifies that the element blocks named in the blockList be omitted from the analysis.

Description

If an element block is omitted, then it is illegal to refer to it later in the input file e.g an initial condition may not be specified on an omitted element block. The elements, faces, etc are never created and it is as if the omitted element blocks did not exist in the mesh file. If a surface is completely determined by the omitted element block, then it is illegal to specify boundary conditions on that surface. However, if the surface spans multiple element blocks, boundary conditions may be applied on the portion of the surface supported by the element blocks that are not omitted.

Parameter	Value	Default
BlockList	string. . .	–

Omit Volume

Syntax

Omit Volume *VolumeList. . .*

Summary

Specifies that the volumes named in the volumeList be omitted from the analysis.

Description

If a volume is omitted, then it is illegal to refer to it later in the input file e.g an initial condition may not be specified on an omitted volume. The elements, faces, etc are never created and it is as if the omitted volumes did not exist in the mesh file. If a surface is completely determined by the omitted volume, then it is illegal to specify boundary conditions on that surface. However, if the surface spans multiple volumes, boundary conditions may be applied on the portion of the surface supported by the volumes that are not omitted.

Parameter	Value	Default
VolumeList	string. . .	–

Time Scale Factor

Syntax

Time Scale Factor *{=}* *Scale*

Summary

The scale factor to be applied to the times on the mesh database. If the scale factor is 20 and the times on the mesh database are 0.1, 0.2, 0.3, then the application will see the mesh times as 2, 4, 6.

Parameter	Value	Default
{=}	{= | is}	–
Scale	real	–

Use Generic Names

Syntax

Use Generic Names

Summary

If this command is present then the name of all blocks and sets in the mesh will be of the form “type_”+id. For example, an element block with id=42 will be named “block_42”; a sideset with id 314 will be named “surface_314”. If there are any names in the mesh file, those names will be aliases for the blocks and sets. If this command is not present, then if a name is in the mesh file, it will be used as the name and the generic generated name will be an alias. This is used as a workaround in codes that do not correctly handle named blocks and sets or as a workaround in meshes which contain non-user-specified names.

Use Material

Syntax

Use Material *MaterialName* For *VolumeList. . .*

Summary

Associate the given volumes with the indicated material name.

Parameter	Value	Default
MaterialName	string	–
VolumeList	string. . .	–

Coordinate System

Syntax

Coordinate System *{=} {axisymmetric | barycentric | cartesian | cyclidic | cylindrical | polar | quadriplanar | skew | spherical | toroidal | trilinear}*

Summary

The interpretation of the geometry data stored in this database. Optional. Defaults to Cartesian.

Parameter	Value	Default
{=}	{= | are | is}	–
CoordinateSystem	{axisymmetric | barycentric | cartesian | cyclidic | cylindrical | polar | quadriplanar | skew | spherical | toroidal | trilinear}	–

Decomposition Method

Syntax

Decomposition Method *{=} {block | cyclic | external | geom_kway | hsfc | kway | kway_geom | linear | map | metis_sfc | random | rcb | rib | variable}*

Summary

The decomposition algorithm to be used to partition elements to each processor in a parallel run.

Parameter	Value	Default
{=}	{= | are | is}	–
Method	{block | cyclic | external | geom_kway | hsfc | kway | kway_geom | linear | map | metis_sfc | random | rcb | rib | variable}	–

Assembly

Scope

Finite Element Model

Summary

This Block allows the definition of an assembly in the input file.

```
begin Assembly Assembly_Name

   Allow Missing Block Ids {=} {off | on}

   Assembly {=} AssemblyName...

   Block {=} BlockName...

   Block_Id List {=} BlockIndex...

   Nodeset {=} NodesetName...

   Nodeset_Id List {=} BlockIndex...

   Surface {=} SurfaceName...

   Surface_Id List {=} BlockIndex...

end Assembly Assembly_Name
```

Line Commands

Allow Missing Block Ids

Syntax

Allow Missing Block Ids *{=} {off | on}*

Summary

Permits the 'Block_Id List' command to have invalid block identifiers. Invalid identifiers will not be reported.

Parameter	Value	Default
{=}	{= | are | is}	–
AllowMissingBlockIds	{off | on}	–

Assembly

Syntax

Assembly *{=} AssemblyName. . .*

Summary

Assemblies that are to be added to an assembly.

Parameter	Value	Default
{=}	{= | are | is}	–
AssemblyName	string. . .	–

Block

Syntax

Block *{=} BlockName. . .*

Summary

Blocks that are to be added to an assembly.

Parameter	Value	Default
{=}	{= | are | is}	–
BlockName	string. . .	–

Block_Id List

Syntax

Block_Id List *{=} BlockIndex. . .*

Summary

Add list of block ids to assemblies using ID, FirstID:LastId, or FistID:LastID:Increment

Parameter	Value	Default
{=}	{= | are | is}	–
BlockIndex	string. . .	–

Nodeset

Syntax

Nodeset *{=} NodesetName. . .*

Summary

Nodesets that are to be added to an assembly.

Parameter	Value	Default
{=}	{= | are | is}	–
NodesetName	string. . .	–

Nodeset_Id List

Syntax

Nodeset_Id List *{=}* *BlockIndex. . .*

Summary

Add list of nodeset ids to assemblies using ID, FirstID:LastId, or FistID:LastID:Increment

Parameter	Value	Default
{=}	{= | are | is}	–
BlockIndex	string. . .	–

Surface

Syntax

Surface *{=}* *SurfaceName. . .*

Summary

Surfaces that are to be added to an assembly.

Parameter	Value	Default
{=}	{= | are | is}	–
SurfaceName	string. . .	–

Surface_Id List

Syntax

Surface_Id List *{=}* *BlockIndex. . .*

Summary

Add list of surface ids to assemblies using ID, FirstID:LastId, or FistID:LastID:Increment

Parameter	Value	Default
{=}	{= | are | is}	–
BlockIndex	string. . .	–

Decomposition Method

Syntax

Decomposition Method *{=} {block | cyclic | external | geom_kway | hsfc | kway | kway_geom | linear | map | metis_sfc | random | rcb | rib | variable}*

Summary

The decomposition algorithm to be used to partition elements to each processor in a parallel run.

Parameter	Value	Default
{=}	{= | are | is}	–
Method	{block | cyclic | external | geom_kway | hsfc | kway | kway_geom | linear | map | metis_sfc | random | rcb | rib | variable}	–

List Mesh Groups

Syntax

List Mesh Groups [*Name. . .*]

Summary

This command is deprecated and no longer has any effect. Mesh group summaries are always printed to the log file.

Material

Syntax

Material *MatName*

Summary

Associates this element block with its material properties.

Parameter	Value	Default
MatName	string	–

Material =

Syntax

Material = *MatName*

Summary

Associates this element block with its material properties.

Parameter	Value	Default
MatName	string	–

Mesh Group

Syntax

Mesh Group *name {=} Mesh_extent. . .*

Summary

Specifies a user selected grouping for mesh entities of the same type to be grouped under a single name alias. While defining a simulation model the alias can be used when referring to the entire collection of mesh entities. By default the group name ALL_BLOCKS is always supplied.

Parameter	Value	Default
name	string	–
{=}	{= | are | is}	–
Mesh_extent	string. . .	all_blocks

7.4 IO Region

Input_Output Region

Scope

Procedure

Summary

An input output region can load data from an exodus file and transfer it to other regions, either constant in time or varying. It can be used for setting initial conditions from prior data, defining boundary conditions, or any other applications of custom field values.

For output it can be used to visualize data on a reduced set of the domain - for example one could transfer to an input output region that had a 2D slice through a 3D domain, or a coarser 3D mesh.

```
begin Input_Output Region Parameter_block_name

  Create Element Field Field_name Of Type {asym_tensor_03 | complex |␣
↪full_tensor_22 | full_tensor_36 | integer | long_integer | matrix_22␣
↪| matrix_33 | real | sym_tensor_21 | sym_tensor_31 | sym_tensor_33 |␣
↪unsigned_integer | unsigned_integer_64 | vector_2d | vector_3d} And␣
↪Dimension Dimension [ Value {=} Number...  ]

  Create Nodal Field Field_name Of Type {asym_tensor_03 | complex |␣
↪full_tensor_22 | full_tensor_36 | integer | long_integer | matrix_22␣
↪| matrix_33 | real | sym_tensor_21 | sym_tensor_31 | sym_tensor_33 |␣
↪unsigned_integer | unsigned_integer_64 | vector_2d | vector_3d} And␣
↪Dimension Dimension [ Value {=} Number...  ]

  Fixed Time [ {=} Fixed_time  ]

  Offset Time {=} Period_offset_time

  Periodicity Time {=} Periodicity_time

  Start Time {=} Start_time

  Time Interpolation Method {=} {closest | linear}

  Timestep Adjustment Interval {=} Nsteps

  Use Finite Element Model ModelName [ Model Coordinates Are Nodal_
↪variable_name  ]

  begin Heartbeat Label
  end

  begin Heartbeat Output Label
  end

  begin History Output Label
  end

  begin Restart Data Label
  end

  begin Results Output Label
  end

end Input_Output Region Parameter_block_name
```

Line Commands

Create Element Field

Syntax

Create Element Field *Field_name* Of Type *{asym_tensor_03 | complex | full_tensor_22 | full_tensor_36 | integer | long_integer | matrix_22 | matrix_33 | real | sym_tensor_21 | sym_tensor_31 | sym_tensor_33 | unsigned_integer | unsigned_integer_64 | vector_2d | vector_3d}* And Dimension *Dimension* [*Value {=} Number. . .*]

Summary

Creates a Element Field name field_name on the region.

Parameter	Value	Default
Field_name	string	–
Option	{asym_tensor_03 | complex | full_tensor_22 | full_tensor_36 | integer | long_integer | matrix_22 | matrix_33 | real | sym_tensor_21 | sym_tensor_31 | sym_tensor_33 | unsigned_integer | unsigned_integer_64 | vector_2d | vector_3d}	–
Dimension	integer	–

Create Nodal Field

Syntax

Create Nodal Field *Field_name* Of Type *{asym_tensor_03 | complex | full_tensor_22 | full_tensor_36 | integer | long_integer | matrix_22 | matrix_33 | real | sym_tensor_21 | sym_tensor_31 | sym_tensor_33 | unsigned_integer | unsigned_integer_64 | vector_2d | vector_3d}* And Dimension *Dimension* [*Value {=} Number. . .*]

Summary

Creates a Nodal Field name field_name on the region.

Parameter	Value	Default
Field_name	string	–
Option	{asym_tensor_03 | complex | full_tensor_22 | full_tensor_36 | integer | long_integer | matrix_22 | matrix_33 | real | sym_tensor_21 | sym_tensor_31 | sym_tensor_33 | unsigned_integer | unsigned_integer_64 | vector_2d | vector_3d}	–
Dimension	integer	–

Fixed Time

Syntax

Fixed Time [*{=}* *Fixed_time*]

Summary

The line specifies that the database will be read for a single, fixed time. Specifying the actual time is optional. If the time is not specified, the final time plane in the database will be read.

NOTE:

- This option takes precedence over the periodic specifications given by START TIME, PERIODICITY TIME, and OFFSET TIME.

```
if FIXED TIME is specified then
   if FIXED TIME value is given then (eg., FIXED TIME is 1.)
      DATABASE TIME = FIXED TIME
   else (eg., FIXED TIME)
      DATABASE TIME = last time in database
else
   if PERIODICITY TIME greater than 0 then
      if APPLICATION TIME less than or equal to START TIME␣
↪then
         DATABASE TIME = APPLICATION TIME
      else
         DATABASE TIME = START TIME +
          (APPLICATION TIME - START TIME) modulo PERIODICITY␣
↪TIME
   else
      DATABASE TIME = APPLICATION TIME
   now add OFFSET TIME to the computed DATABASE TIME
```

Offset Time

Syntax

Offset Time *{=}* *Period_offset_time*

Summary

This value is added to the application time to determine what database time slice to input. If OFFSET TIME were 15 than at application time 0 database time slice 15 would be read from the file and used for the initial values. At application time 1, database time slice 16 would be read.

NOTES:

- The OFFSET TIME is added in after the START TIME and PERIODICITY TIME are used.
- The FIXED TIME option takes precedence over this option.

```
if FIXED TIME is specified then
   if FIXED TIME value is given then (eg., FIXED TIME is 1.)
      DATABASE TIME = FIXED TIME
   else (eg., FIXED TIME)
      DATABASE TIME = last time in database
else
   if PERIODICITY TIME greater than 0 then
      if APPLICATION TIME less than or equal to START TIME␣
↪then
         DATABASE TIME = APPLICATION TIME
      else
         DATABASE TIME = START TIME +
          (APPLICATION TIME - START TIME) modulo PERIODICITY␣
↪TIME
   else
      DATABASE TIME = APPLICATION TIME
   now add OFFSET TIME to the computed DATABASE TIME
```

Parameter	Value	Default
{=}	{= | are | is}	–
Period_offset_time	real	–

Periodicity Time

Syntax

Periodicity Time *{=}* *Periodicity_time*

Summary

START TIME and PERIODICITY TIME taken together give the time frame from the input database to use to initialize the application values. If START TIME is 25 and PERIODICITY TIME is 10, then time slices from 25 to 35 will

be used over and over again as the application time runs from 0 to whatever. In general

```
DATABASE TIME is (APPLICATION TIME - START TIME) modulo␣
↪PERIODICITY TIME
```

after the application time reaches the START TIME.

NOTES:

- The OFFSET TIME is added in after the START TIME and PERIODICITY TIME are used.
- The FIXED TIME option takes precedence over this option.

```
if FIXED TIME is specified then
   if FIXED TIME value is given then (eg., FIXED TIME is 1.)
      DATABASE TIME = FIXED TIME
   else (eg., FIXED TIME)
      DATABASE TIME = last time in database
else
   if PERIODICITY TIME greater than 0 then
      if APPLICATION TIME less than or equal to START TIME␣
↪then
         DATABASE TIME = APPLICATION TIME
      else
         DATABASE TIME = START TIME +
          (APPLICATION TIME - START TIME) modulo PERIODICITY␣
↪TIME
   else
      DATABASE TIME = APPLICATION TIME
   now add OFFSET TIME to the computed DATABASE TIME
```

Parameter	Value	Default
{=}	{= | are | is}	–
Periodicity_time	real	–

Start Time

Syntax

Start Time *{=}* *Start_time*

Summary

The time in which to start applying PERIODICITY TIME. If PERIODICITY TIME is not specified then START TIME is ignored.

NOTES:

- The OFFSET TIME is added in after the START TIME and PERIODICITY TIME are used.
- The FIXED TIME option takes precedence over this option.

```
if FIXED TIME is specified then
   if FIXED TIME value is given then (eg., FIXED TIME is 1.)
      DATABASE TIME = FIXED TIME
   else (eg., FIXED TIME)
      DATABASE TIME = last time in database
else
   if PERIODICITY TIME greater than 0 then
      if APPLICATION TIME less than or equal to START TIME␣
↪then
         DATABASE TIME = APPLICATION TIME
      else
         DATABASE TIME = START TIME +
          (APPLICATION TIME - START TIME) modulo PERIODICITY␣
↪TIME
   else
      DATABASE TIME = APPLICATION TIME
   now add OFFSET TIME to the computed DATABASE TIME
```

Parameter	Value	Default
{=}	{= | are | is}	–
Start_time	real	–

Time Interpolation Method

Syntax

Time Interpolation Method *{=} {closest | linear}*

Summary

This line specifies how interpolation in time in the database will be handled. If linear (the default option) is specified, quantities at a given point are linearly interpolated from the bounding known time points. If the closest option is selected, then the closest known time point will be taken.

Parameter	Value	Default
{=}	{= | are | is}	–
Method	{closest | linear}	–

Timestep Adjustment Interval

Syntax

Timestep Adjustment Interval *{=} Nsteps*

Summary

Specify the number of steps to 'look ahead' and adjust the timestep to ensure that the specified output times or simulation end time will be hit 'exactly'.

Parameter	Value	Default
{=}	{= | are | is}	–
Nsteps	integer	–

Use Finite Element Model

Syntax

Use Finite Element Model *ModelName* [Model Coordinates Are *Nodal_variable_name*]

Summary

Associates a predefined finite element model with this region.

Parameter	Value	Default
ModelName	string	–

7.5 Linear Solvers

Tpetra

Tpetra Equation Solver

Scope

Sierra

Summary

Use a Tpetra-based linear solver.

Description

If the linear solve fails it will report an error code. While these codes are not universally standardized across linear solvers, they can generally be interpreted with the following rules:

- **maxits**: The linear solver reached the maximum number of linear solver iterations and did not reach a solution. Check the final linear solver residual to see if this is serious or not. Even if the error is not serious (residuals are small) this usually means the solve is much more expensive than necessary and a better solver/preconditioner combination should be used.

- **Error 1 (or -1)**: The linear solver encountered a recoverable internal error. As always, check the final linear solver residual reported in the log file to determine if it is serious or not.
- **Error 2 (or -2, 4, -4, 5, -5)**: The linear solver encountered a fatal internal error. This is often because of a singular matrix or a NaN or Infinite term. This is almost always a serious error.
- **Error 3 (or -3)**: The linear solver had an internal loss of precision. This is often not a serious error, but may indicate the need for a better solver/preconditioner combination. As always, check the final linear solver residual reported in the log file to determine if it is serious or not.

```
begin Tpetra Equation Solver Solver Name

  Bc Enforcement {=} {exact | exact_no_column_mod | remove | solver |␣
↪solver_no_column_mod}

  Failed Solve Matrix Output {=} value

  Matrix Output {=} value

  Matrix Output Time Range {=} value1[ value2]

  Matrix Scaling {=} {diagonal | one_norm}

  begin Bicgstab Solver
  end

  begin Cg Solver
  end

  begin Expert Belos Solver
  end

  begin Gcrodr Solver
  end

  begin Gmres Solver
  end

  begin Klu2 Solver
  end

  begin Poly Solver
  end
```

```
begin Preset Solver
end

begin Preset Solver
end

begin Superlu Solver
end

begin Umfpack Solver
end
```

end Tpetra Equation Solver *Solver Name*

Line Commands

Bc Enforcement

Syntax

Bc Enforcement *{=} {exact | exact_no_column_mod | remove | solver | solver_no_column_mod}*

Summary

Control how Dirichlet BC's are enforced. Fuego only.

Parameter	Value	Default
{=}	{= | are | is}	–
value	{exact | exact_no_column_mod | remove | solver | solver_no_column_mod}	–

Failed Solve Matrix Output

Syntax

Failed Solve Matrix Output *{=} value*

Summary

On a failed linear solve, output the matrix to the specified file name.

Parameter	Value	Default
{=}	{= | are | is}	–
value	string	–

Matrix Output

Syntax

Matrix Output *{=} value*

Summary

Output the matrix to the specified file name.

Parameter	Value	Default
{=}	{= | are | is}	–
value	string	–

Matrix Output Time Range

Syntax

Matrix Output Time Range *{=} value1[value2]*

Summary

Range of simulation time to output the matrix to file, default is dump out all matrices per step.

Parameter	Value	Default
{=}	{= | are | is}	–
value	real1[real2]	–

Matrix Scaling

Syntax

Matrix Scaling *{=} {diagonal | one_norm}*

Summary

Apply a pre-solve scaling to the matrix.

Description

In some problems scaling the matrix prior to the linear solve can help reduce the condition number of the matrix. This should be used in addition to an appropriate preconditioner.

Parameter	Value	Default
{=}	{= | are | is}	–
value	{diagonal | one_norm}	–

Solvers

Klu2 Solver

Scope
Teko Subblock Solver, Tpetra Equation Solver

Summary
Use a KLU2 direct solver.

```
begin Klu2 Solver

end Klu2 Solver
```

Superlu Solver

Scope
Teko Subblock Solver, Tpetra Equation Solver

Summary
Use a SUPERLU direct solver.

```
begin Superlu Solver

end Superlu Solver
```

Umfpack Solver

Scope
Teko Subblock Solver, Tpetra Equation Solver

Summary
Use a UMFPACK direct solver.

```
begin Umfpack Solver

end Umfpack Solver
```

Cg Solver

Scope

Teko Subblock Solver, Tpetra Equation Solver

Summary

Use a conjugate gradient solver.

```
begin Cg Solver

  Convergence Tolerance {=} value [ Minimum = tolerance_floor  ]

  Maximum Iterations {=} value

  Residual Output {=} [ {disabled | enabled}  ]

  Residual Scaling {=} {none | preconditioned_r0 | r0 | rhs}

  begin Dd-Ilu Preconditioner
  end

  begin Dd-Ilut Preconditioner
  end

  begin Dd-Parilut Preconditioner
  end

  begin Expert Ifpack2 Preconditioner
  end

  begin Fastilu Preconditioner
  end

  begin Jacobi Preconditioner
  end

  begin Muelu Preconditioner
  end

  begin Preset Preconditioner
  end

  begin Sgs Preconditioner
  end
```

```
   begin Sgs2 Preconditioner
   end

end Cg Solver
```

Line Commands

Convergence Tolerance

Syntax

Convergence Tolerance *{=}* *value* [Minimum = *tolerance_floor*]

Summary

Set the convergence tolerance for the linear solver. Setting CONVERGENCE TOLERANCE = AUTOMATIC enables an autonomous linear convergence tolerance calculation, which is computed from a combination of the initial linear residual and user-specified nonlinear residual tolerance. An additional floor value on the linear convergence tolerance can be declared with CONVERGENCE TOLERANCE = AUTOMATIC MINIMUM = <val>, where the default floor value is 1.0e-6

Parameter	Value	Default
{=}	{= | are | is}	–
value	string	–

Maximum Iterations

Syntax

Maximum Iterations *{=}* *value*

Summary

Set the maximum number of iterations taken by the linear solver

Parameter	Value	Default
{=}	{= | are | is}	–
value	integer	300

Residual Output

Syntax

Residual Output *{=}* [*{disabled | enabled}*]

Summary

Output the linear residual history to files in the residual_output/ directory, which is created automatically. For timestep NNN and linear system NAME, this will output a .csv file named residual_output/linear_solves_NAME.stepNNN. This

file includes information about the linear residual at every iteration across all nonlinear iterations.

Parameter	Value	Default
{=}	{= | are | is}	–

Residual Scaling

Syntax

Residual Scaling *{=} {none | preconditioned_r0 | r0 | rhs}*

Summary

Set the residual scaling for the linear solver

Parameter	Value	Default
{=}	{= | are | is}	–
value	{none | preconditioned_r0 | r0 | rhs}	r0

Bicgstab Solver

Scope

Teko Subblock Solver, Tpetra Equation Solver

Summary

Use a bicgstab solver.

```
begin Bicgstab Solver

   Convergence Tolerance {=} value [ Minimum = tolerance_floor  ]

   Maximum Iterations {=} value

   Residual Output {=} [ {disabled | enabled}  ]

   Residual Scaling {=} {none | preconditioned_r0 | r0 | rhs}

   begin Dd-Ilu Preconditioner
   end

   begin Dd-Ilut Preconditioner
   end

   begin Dd-Parilut Preconditioner
   end
```

```
  begin Expert Ifpack2 Preconditioner
  end

  begin Fastilu Preconditioner
  end

  begin Jacobi Preconditioner
  end

  begin Muelu Preconditioner
  end

  begin Preset Preconditioner
  end

  begin Sgs Preconditioner
  end

  begin Sgs2 Preconditioner
  end

end Bicgstab Solver
```

Line Commands

Convergence Tolerance

Syntax

Convergence Tolerance *{=}* *value* [Minimum = *tolerance_floor*]

Summary

Set the convergence tolerance for the linear solver. Setting CONVERGENCE TOLERANCE = AUTOMATIC enables an autonomous linear convergence tolerance calculation, which is computed from a combination of the initial linear residual and user-specified nonlinear residual tolerance. An additional floor value on the linear convergence tolerance can be declared with CONVERGENCE TOLERANCE = AUTOMATIC MINIMUM = <val>, where the default floor value is 1.0e-6

Parameter	Value	Default
{=}	{= | are | is}	–
value	string	–

Maximum Iterations

Syntax

Maximum Iterations *{=}* *value*

Summary

Set the maximum number of iterations taken by the linear solver

Parameter	Value	Default
{=}	{= | are | is}	–
value	integer	300

Residual Output

Syntax

Residual Output *{=}* [*{disabled | enabled}*]

Summary

Output the linear residual history to files in the residual_output/ directory, which is created automatically. For timestep NNN and linear system NAME, this will output a .csv file named residual_output/linear_solves_NAME.stepNNN. This file includes information about the linear residual at every iteration across all nonlinear iterations.

Parameter	Value	Default
{=}	{= | are | is}	–

Residual Scaling

Syntax

Residual Scaling *{=}* *{none | preconditioned_r0 | r0 | rhs}*

Summary

Set the residual scaling for the linear solver

Parameter	Value	Default
{=}	{= | are | is}	–
value	{none | preconditioned_r0 | r0 | rhs}	r0

Gmres Solver

Scope

Teko Block Solver, Teko Subblock Solver, Tpetra Equation Solver

Summary

Use a GMRES solver.

begin Gmres Solver

Convergence Tolerance *{=}* *value* [Minimum = *tolerance_floor*]

Maximum Iterations *{=}* *value*

Residual Output *{=}* [*{disabled | enabled}*]

Residual Scaling *{=}* *{none | preconditioned_r0 | r0 | rhs}*

Restart Iterations *{=}* *value*

begin Dd-Ilu Preconditioner
end

begin Dd-Ilut Preconditioner
end

begin Dd-Parilut Preconditioner
end

begin Expert Ifpack2 Preconditioner
end

begin Expert Teko Preconditioner
end

begin Fastilu Preconditioner
end

begin Jacobi Preconditioner
end

begin Muelu Preconditioner
end

begin Preset Preconditioner
end

begin Sgs Preconditioner
end

begin Sgs2 Preconditioner
end

```
begin Teko Preconditioner
end
```

```
end Gmres Solver
```

Line Commands

Convergence Tolerance

Syntax

Convergence Tolerance *{=}* *value* [Minimum = *tolerance_floor*]

Summary

Set the convergence tolerance for the linear solver. Setting CONVERGENCE TOLERANCE = AUTOMATIC enables an autonomous linear convergence tolerance calculation, which is computed from a combination of the initial linear residual and user-specified nonlinear residual tolerance. An additional floor value on the linear convergence tolerance can be declared with CONVERGENCE TOLERANCE = AUTOMATIC MINIMUM = <val>, where the default floor value is 1.0e-6

Parameter	Value	Default
{=}	{= | are | is}	–
value	string	–

Maximum Iterations

Syntax

Maximum Iterations *{=}* *value*

Summary

Set the maximum number of iterations taken by the linear solver

Parameter	Value	Default
{=}	{= | are | is}	–
value	integer	300

Residual Output

Syntax

Residual Output *{=}* [*{disabled | enabled}*]

Summary

Output the linear residual history to files in the residual_output/ directory, which is created automatically. For timestep NNN and linear system NAME, this will output a .csv file named residual_output/linear_solves_NAME.stepNNN. This

file includes information about the linear residual at every iteration across all nonlinear iterations.

Parameter	Value	Default
{=}	{= | are | is}	–

Residual Scaling

Syntax

Residual Scaling *{=} {none | preconditioned_r0 | r0 | rhs}*

Summary

Set the residual scaling for the linear solver

Parameter	Value	Default
{=}	{= | are | is}	–
value	{none | preconditioned_r0 | r0 | rhs}	r0

Restart Iterations

Syntax

Restart Iterations *{=} value*

Summary

Set the number of iterations between GMRES restarts.

Parameter	Value	Default
{=}	{= | are | is}	–
value	integer	30

Poly Solver

Scope

Teko Subblock Solver, Tpetra Equation Solver

Summary

Use a poly solver.

```
begin Poly Solver

  Convergence Tolerance {=} value [ Minimum = tolerance_floor  ]

  Maximum Iterations {=} value

  Polynomial Order {=} value
```

```
   Residual Output {=} [ {disabled | enabled}  ]

   Residual Scaling {=} {none | preconditioned_r0 | r0 | rhs}

   Restart Iterations {=} value

   begin Dd-Ilu Preconditioner
   end

   begin Dd-Ilut Preconditioner
   end

   begin Dd-Parilut Preconditioner
   end

   begin Expert Ifpack2 Preconditioner
   end

   begin Fastilu Preconditioner
   end

   begin Jacobi Preconditioner
   end

   begin Muelu Preconditioner
   end

   begin Preset Preconditioner
   end

   begin Sgs Preconditioner
   end

   begin Sgs2 Preconditioner
   end

end Poly Solver
```

Line Commands

Convergence Tolerance

Syntax

Convergence Tolerance *{=}* *value* [Minimum = *tolerance_floor*]

Summary

Set the convergence tolerance for the linear solver. Setting CONVERGENCE TOLERANCE = AUTOMATIC enables an autonomous linear convergence tolerance calculation, which is computed from a combination of the initial linear residual and user-specified nonlinear residual tolerance. An additional floor value on the linear convergence tolerance can be declared with CONVERGENCE TOLERANCE = AUTOMATIC MINIMUM = <val>, where the default floor value is 1.0e-6

Parameter	Value	Default
{=}	{= | are | is}	–
value	string	–

Maximum Iterations

Syntax

Maximum Iterations *{=}* *value*

Summary

Set the maximum number of iterations taken by the linear solver

Parameter	Value	Default
{=}	{= | are | is}	–
value	integer	300

Polynomial Order

Syntax

Polynomial Order *{=}* *value*

Summary

Set the polynomial order of polynomial preconditioner.

Parameter	Value	Default
{=}	{= | are | is}	–
value	integer	5

Residual Output

Syntax

Residual Output *{=}* [*{disabled | enabled}*]

Summary

Output the linear residual history to files in the residual_output/ directory, which is created automatically. For timestep NNN and linear system NAME, this will output a .csv file named residual_output/linear_solves_NAME.stepNNN. This file includes information about the linear residual at every iteration across all nonlinear iterations.

Parameter	Value	Default
{=}	{= | are | is}	–

Residual Scaling

Syntax

Residual Scaling *{=} {none | preconditioned_r0 | r0 | rhs}*

Summary

Set the residual scaling for the linear solver

Parameter	Value	Default
{=}	{= | are | is}	–
value	{none | preconditioned_r0 | r0 | rhs}	r0

Restart Iterations

Syntax

Restart Iterations *{=} value*

Summary

Set the number of iterations between GMRES restarts.

Parameter	Value	Default
{=}	{= | are | is}	–
value	integer	30

Gcrodr Solver

Scope

Teko Subblock Solver, Tpetra Equation Solver

Summary

Use a GCRODR solver.

```
begin Gcrodr Solver

   Convergence Tolerance {=} value [ Minimum = tolerance_floor  ]

   Maximum Iterations {=} value

   Recycled Blocks {=} value

   Residual Output {=} [ {disabled | enabled}  ]

   Residual Scaling {=} {none | preconditioned_r0 | r0 | rhs}

   Restart Iterations {=} value

   begin Dd-Ilu Preconditioner
   end

   begin Dd-Ilut Preconditioner
   end

   begin Dd-Parilut Preconditioner
   end

   begin Expert Ifpack2 Preconditioner
   end

   begin Fastilu Preconditioner
   end

   begin Jacobi Preconditioner
   end

   begin Muelu Preconditioner
   end

   begin Preset Preconditioner
   end

   begin Sgs Preconditioner
   end

   begin Sgs2 Preconditioner
   end

end Gcrodr Solver
```

Line Commands

Convergence Tolerance

Syntax

Convergence Tolerance *{=}* *value* [Minimum = *tolerance_floor*]

Summary

Set the convergence tolerance for the linear solver. Setting CONVERGENCE TOLERANCE = AUTOMATIC enables an autonomous linear convergence tolerance calculation, which is computed from a combination of the initial linear residual and user-specified nonlinear residual tolerance. An additional floor value on the linear convergence tolerance can be declared with CONVERGENCE TOLERANCE = AUTOMATIC MINIMUM = <val>, where the default floor value is 1.0e-6

Parameter	Value	Default
{=}	{= | are | is}	–
value	string	–

Maximum Iterations

Syntax

Maximum Iterations *{=}* *value*

Summary

Set the maximum number of iterations taken by the linear solver

Parameter	Value	Default
{=}	{= | are | is}	–
value	integer	300

Recycled Blocks

Syntax

Recycled Blocks *{=}* *value*

Summary

Set the number of recycle blocks in GCRODR.

Parameter	Value	Default
{=}	{= | are | is}	–
value	integer	5

Residual Output

Syntax

Residual Output *{=}* [*{disabled | enabled}*]

Summary

Output the linear residual history to files in the residual_output/ directory, which is created automatically. For timestep NNN and linear system NAME, this will output a .csv file named residual_output/linear_solves_NAME.stepNNN. This file includes information about the linear residual at every iteration across all nonlinear iterations.

Parameter	Value	Default
{=}	{= | are | is}	–

Residual Scaling

Syntax

Residual Scaling *{=} {none | preconditioned_r0 | r0 | rhs}*

Summary

Set the residual scaling for the linear solver

Parameter	Value	Default
{=}	{= | are | is}	–
value	{none | preconditioned_r0 | r0 | rhs}	r0

Restart Iterations

Syntax

Restart Iterations *{=} value*

Summary

Set the number of iterations between GCRODR restarts.

Parameter	Value	Default
{=}	{= | are | is}	–
value	integer	30

Expert Belos Solver

Scope

Teko Block Solver, Teko Subblock Solver, Tpetra Equation Solver

Summary

Use a Belos solver with parameters specified in an XML file.

begin Expert Belos Solver

Xml File *{=}* *value*

begin Dd-Ilu Preconditioner
end

begin Dd-Ilut Preconditioner
end

begin Dd-Parilut Preconditioner
end

begin Expert Ifpack2 Preconditioner
end

begin Expert Teko Preconditioner
end

begin Fastilu Preconditioner
end

begin Jacobi Preconditioner
end

begin Muelu Preconditioner
end

begin Preset Preconditioner
end

begin Sgs Preconditioner
end

begin Sgs2 Preconditioner
end

begin Teko Preconditioner
end

end Expert Belos Solver

Line Commands

Xml File

Syntax

Xml File *{=}* *value*

Summary

Supply the name of an XML file containing a Belos parameter list.

Description

The supplied XML file must contain a ParameterList with a “solution method” string parameter corresponding to the desired type of Belos solver. Any additional parameters that should be passed on to Belos should be part of a nested ParameterList with name=”Belos-Tpetra”.

Parameter	Value	Default
{=}	{= | are | is}	–
value	string	–

Preset Solver

Scope

Teko Subblock Solver, Tpetra Equation Solver

Summary

Use a preset solver for a specific linear system type.

Description

The preset solver for a given linear system type is intended to provide reasonable behavior for many problems of that type. If this proves to be insufficient, the user must provide more fine-grained control over solver settings. The equivalent command which produces the solver will be output to the log file, and this may be used as a starting point from which changes for robustness may be made (i.e. increasing GMRES subspace size or changing the preconditioner).

```
begin Preset Solver

   Convergence Tolerance {=} value [ Minimum = tolerance_floor  ]

   Maximum Iterations {=} value

   Residual Output {=} [ {disabled | enabled}  ]

   Residual Scaling {=} {none | preconditioned_r0 | r0 | rhs}
```

```
  Solver Type {=} {continuity | displacement_multigrid | high_aspect_
↪continuity | multiphysics | scalar_transport | teko_multiphysics | teko_
↪ns | thermal | thermal_bicgstab | thermal_multigrid | thermal_symmetric}

end Preset Solver
```

Line Commands

Convergence Tolerance

Syntax

Convergence Tolerance *{=}* *value* [Minimum = *tolerance_floor*]

Summary

Set the convergence tolerance for the linear solver. Setting CONVERGENCE TOLERANCE = AUTOMATIC enables an autonomous linear convergence tolerance calculation, which is computed from a combination of the initial linear residual and user-specified nonlinear residual tolerance. An additional floor value on the linear convergence tolerance can be declared with CONVERGENCE TOLERANCE = AUTOMATIC MINIMUM = <val>, where the default floor value is 1.0e-6

Parameter	Value	Default
{=}	{= | are | is}	–
value	string	–

Maximum Iterations

Syntax

Maximum Iterations *{=}* *value*

Summary

Set the maximum number of iterations taken by the linear solver

Parameter	Value	Default
{=}	{= | are | is}	–
value	integer	300

Residual Output

Syntax

Residual Output *{=}* [*{disabled | enabled}*]

Summary

Output the linear residual history to files in the residual_output/ directory, which is created automatically. For timestep NNN and linear system NAME, this will

output a .csv file named residual_output/linear_solves_NAME.stepNNN. This file includes information about the linear residual at every iteration across all nonlinear iterations.

Parameter	Value	Default
{=}	{= | are | is}	–

Residual Scaling

Syntax

Residual Scaling *{=} {none | preconditioned_r0 | r0 | rhs}*

Summary

Set the residual scaling for the linear solver

Parameter	Value	Default
{=}	{= | are | is}	–
value	{none | preconditioned_r0 | r0 | rhs}	r0

Solver Type

Syntax

Solver Type *{=} {continuity | displacement_multigrid | high_aspect_continuity | multiphysics | scalar_transport | teko_multiphysics | teko_ns | thermal | thermal_bicgstab | thermal_multigrid | thermal_symmetric}*

Summary

Set the preset solver type

Parameter	Value	Default
{=}	{= | are | is}	–
value	{continuity | displacement_multigrid | high_aspect_continuity | multiphysics | scalar_transport | teko_multiphysics | teko_ns | thermal | thermal_bicgstab | thermal_multigrid | thermal_symmetric}	–

Preset Solver

Scope

Teko Block Solver, Teko Subblock Solver, Tpetra Equation Solver

Summary

Use a preset solver for a specific linear system type.

Description

The preset solver for a given linear system type is intended to provide reasonable behavior for many problems of that type. If this proves to be insufficient, the user must provide more fine-grained control over solver settings. The equivalent command which produces the solver will be output to the log file, and this may be used as a starting point from which changes for robustness may be made (i.e. increasing GMRES subspace size or changing the preconditioner).

```
begin Preset Solver

   Convergence Tolerance {=} value [ Minimum = tolerance_floor  ]

   Maximum Iterations {=} value

   Residual Output {=} [ {disabled | enabled}  ]

   Residual Scaling {=} {none | preconditioned_r0 | r0 | rhs}

   Solver Type {=} {continuity | displacement_multigrid | high_aspect_
↪continuity | multiphysics | scalar_transport | teko_multiphysics | teko_
↪ns | thermal | thermal_bicgstab | thermal_multigrid | thermal_symmetric}

end Preset Solver
```

Line Commands

Convergence Tolerance

Syntax

Convergence Tolerance *{=}* *value* [Minimum = *tolerance_floor*]

Summary

Set the convergence tolerance for the linear solver. Setting CONVERGENCE TOLERANCE = AUTOMATIC enables an autonomous linear convergence tolerance calculation, which is computed from a combination of the initial linear residual and user-specified nonlinear residual tolerance. An additional floor value on the linear convergence tolerance can be declared with

CONVERGENCE TOLERANCE = AUTOMATIC MINIMUM = <val>, where the default floor value is 1.0e-6

Parameter	Value	Default
{=}	{= | are | is}	–
value	string	–

Maximum Iterations

Syntax

Maximum Iterations *{=}* *value*

Summary

Set the maximum number of iterations taken by the linear solver

Parameter	Value	Default
{=}	{= | are | is}	–
value	integer	300

Residual Output

Syntax

Residual Output *{=}* [*{disabled | enabled}*]

Summary

Output the linear residual history to files in the residual_output/ directory, which is created automatically. For timestep NNN and linear system NAME, this will output a .csv file named residual_output/linear_solves_NAME.stepNNN. This file includes information about the linear residual at every iteration across all nonlinear iterations.

Parameter	Value	Default
{=}	{= | are | is}	–

Residual Scaling

Syntax

Residual Scaling *{=}* *{none | preconditioned_r0 | r0 | rhs}*

Summary

Set the residual scaling for the linear solver

Parameter	Value	Default
{=}	{= | are | is}	–
value	{none | preconditioned_r0 | r0 | rhs}	r0

Solver Type

Syntax

Solver Type *{=} {continuity | displacement_multigrid | high_aspect_continuity | multiphysics | scalar_transport | teko_multiphysics | teko_ns | thermal | thermal_bicgstab | thermal_multigrid | thermal_symmetric}*

Summary

Set the preset solver type

Parameter	Value	Default
{=}	{= | are | is}	–
value	{continuity | displacement_multigrid | high_aspect_continuity | multiphysics | scalar_transport | teko_multiphysics | teko_ns | thermal | thermal_bicgstab | thermal_multigrid | thermal_symmetric}	–

Preconditioners

Expert Teko Preconditioner

Scope

Expert Belos Solver, Gmres Solver

Summary

Use a teko preconditioner with parameters specified in an XML file.

```
begin Expert Teko Preconditioner

   Define Matrix Subblock Number = number Dof = dof name

   Inverse Method {=} value

   Xml File {=} value

end Expert Teko Preconditioner
```

Line Commands

Define Matrix Subblock

Syntax

Define Matrix Subblock Number = *number* Dof = *dof name*

Summary

Define the dof associated with a matrix subblock for block preconditioning.

Description

Define the matrix subblock number and the equation to apply to that subblock. Please see https://trilinos.github.io/teko.html for more details

Parameter	Value	Default
number	integer	–
dof name	string	–

Inverse Method

Syntax

Inverse Method *{=}* *value*

Summary

Supply the name of the inverse method.

Description

The name of the inverse method must correspond to the name of a parameter list specified in the teko XML file. Please see https://trilinos.github.io/teko.html for more details

Parameter	Value	Default
{=}	{= | are | is}	–
value	string	–

Xml File

Syntax

Xml File *{=}* *value*

Summary

Supply the name of an XML file containing a teko parameter list.

Description

The supplied XML file must contain a ParameterList with a valid teko parameter list. Please see https://trilinos.github.io/teko.html for more details

Parameter	Value	Default
{=}	{= | are | is}	–
value	string	–

Teko Preconditioner

Scope

Expert Belos Solver, Gmres Solver, Teko Block Preconditioner

Summary

Use a teko preconditioner with parameters specified through the input deck.

```
begin Teko Preconditioner

   Define Matrix Subblock Number = number Dof = dof name

   begin Block Gauss-Seidel Solver
   end

   begin Block Jacobi Solver
   end

   begin Hierarchical Block Gauss-Seidel Solver
   end

   begin Simple Solver
   end

   begin Teko Block Preconditioner Solver Name
   end

   begin Teko Block Solver Solver Name
   end

   begin Teko Subblock Preconditioner Solver Name
   end

   begin Teko Subblock Solver Solver Name
   end

end Teko Preconditioner
```

Line Commands

Define Matrix Subblock

Syntax

Define Matrix Subblock Number = *number* Dof = *dof name*

Summary

Define the dof associated with a matrix subblock for block preconditioning.

Description

Define the matrix subblock number and the equation to apply to that subblock.
Please see https://trilinos.github.io/teko.html for more details

Parameter	Value	Default
number	integer	–
dof name	string	–

Teko Subblock Solver

Scope

Teko Preconditioner

Summary

Define a solver to be used for one or more subblock(s)

```
begin Teko Subblock Solver Solver Name

   begin Bicgstab Solver
   end

   begin Cg Solver
   end

   begin Expert Belos Solver
   end

   begin Gcrodr Solver
   end

   begin Gmres Solver
   end

   begin Klu2 Solver
   end
```

```
   begin Poly Solver
   end

   begin Preset Solver
   end

   begin Preset Solver
   end

   begin Superlu Solver
   end

   begin Umfpack Solver
   end
```

end Teko Subblock Solver *Solver Name*

Teko Subblock Preconditioner

Scope

Teko Preconditioner

Summary

Define a preconditioner to be used to approximate the inverse of one or more subblock(s)

begin Teko Subblock Preconditioner *Solver Name*

```
   begin Dd-Ilu Preconditioner
   end

   begin Dd-Ilut Preconditioner
   end

   begin Dd-Parilut Preconditioner
   end

   begin Expert Ifpack2 Preconditioner
   end

   begin Fastilu Preconditioner
   end

   begin Jacobi Preconditioner
```

```
   end

   begin Muelu Preconditioner
   end

   begin Preset Preconditioner
   end

   begin Sgs Preconditioner
   end

   begin Sgs2 Preconditioner
   end

end Teko Subblock Preconditioner Solver Name
```

Teko Block Solver

Scope

Teko Preconditioner

Summary

Define a solver to be used to approximate the inverse of a set of subblocks in a hierarchical block Gauss-Seidel scheme.

```
begin Teko Block Solver Solver Name

   begin Expert Belos Solver
   end

   begin Gmres Solver
   end

   begin Preset Solver
   end

end Teko Block Solver Solver Name
```

Teko Block Preconditioner

Scope

Teko Preconditioner

Summary

Define a preconditioner to be used to approximate the inverse of a set of subblocks in a hierarchical block Gauss-Seidel scheme.

```
begin Teko Block Preconditioner Solver Name

   begin Preset Preconditioner
   end

   begin Teko Preconditioner
   end

end Teko Block Preconditioner Solver Name
```

Hierarchical Block Gauss-Seidel Solver

Scope

Teko Preconditioner

Summary

Use a Hierarchical Block Gauss-Seidel solver inside a Teko preconditioner.

```
begin Hierarchical Block Gauss-Seidel Solver

   Use Block Inverse block inverse For Hierarchical Block number

   Use Subblocks subblocks... For Hierarchical Block number

   Use Upper Triangle {=} {false | true}

end Hierarchical Block Gauss-Seidel Solver
```

Line Commands

Use Block Inverse

Syntax

Use Block Inverse *block inverse* For Hierarchical Block *number*

Summary

Define the block solver to be used for a given hierarchical block.

Parameter	Value	Default
block inverse	string	–
number	integer	–

Use Subblocks

Syntax

Use Subblocks *subblocks. . .* For Hierarchical Block *number*

Summary

Map subblock(s) to a hierarchical block, which is a set containing one or more blocks.

Parameter	Value	Default
subblocks	integer. . .	–
number	integer	–

Use Upper Triangle

Syntax

Use Upper Triangle *{=} {false | true}*

Summary

Define whether to use upper triangular hierarchical block Gauss-Seidel

Parameter	Value	Default
{=}	{= | are | is}	–
value	{false | true}	FALSE

Muelu Preconditioner

Scope

Bicgstab Solver, Cg Solver, Expert Belos Solver, Gcrodr Solver, Gmres Solver, Poly Solver, Teko Subblock Preconditioner

Summary

Use a MueLu preconditioner.

```
begin Muelu Preconditioner

   Xml File {=} value

end Muelu Preconditioner
```

Line Commands

Xml File

Syntax

Xml File *{=}* *value*

Summary

Supply the name of an XML file containing a MueLu parameter list.

Parameter	Value	Default
{=}	{= | are | is}	–
value	string	–

Jacobi Preconditioner

Scope

Bicgstab Solver, Cg Solver, Expert Belos Solver, Gcrodr Solver, Gmres Solver, Poly Solver, Teko Subblock Preconditioner

Summary

Use a Jacobi preconditioner.

```
begin Jacobi Preconditioner

   Damping Factor {=} value

   Number Of Sweeps {=} value

end Jacobi Preconditioner
```

Line Commands

Damping Factor

Syntax

Damping Factor *{=}* *value*

Summary

Set the relaxation damping factor.

Parameter	Value	Default
{=}	{= | are | is}	–
value	real	1

Number Of Sweeps

Syntax

Number Of Sweeps *{=}* *value*

Summary

Set the number of relaxation sweeps to perform.

Parameter	Value	Default
{=}	{= | are | is}	–
value	integer	3

Sgs2 Preconditioner

Scope

Bicgstab Solver, Cg Solver, Expert Belos Solver, Gcrodr Solver, Gmres Solver, Poly Solver, Teko Subblock Preconditioner

Summary

Use a Two-stage symmetric Gauss-Seidel preconditioner.

```
begin Sgs2 Preconditioner

   Damping Factor {=} value

   Number Of Sweeps {=} value

end Sgs2 Preconditioner
```

Line Commands

Damping Factor

Syntax

Damping Factor *{=}* *value*

Summary

Set the relaxation damping factor.

Parameter	Value	Default
{=}	{= | are | is}	–
value	real	1

Number Of Sweeps

Syntax

Number Of Sweeps *{=}* *value*

Summary

Set the number of relaxation sweeps to perform.

Parameter	Value	Default
{=}	{= | are | is}	–
value	integer	3

Sgs Preconditioner

Scope

Bicgstab Solver, Cg Solver, Expert Belos Solver, Gcrodr Solver, Gmres Solver, Poly Solver, Teko Subblock Preconditioner

Summary

Use a symmetric Gauss-Seidel preconditioner.

```
begin Sgs Preconditioner

   Damping Factor {=} value

   Number Of Sweeps {=} value

end Sgs Preconditioner
```

Line Commands

Damping Factor

Syntax

Damping Factor *{=}* *value*

Summary

Set the relaxation damping factor.

Parameter	Value	Default
{=}	{= | are | is}	–
value	real	1

Number Of Sweeps

Syntax

Number Of Sweeps *{=}* *value*

Summary

Set the number of relaxation sweeps to perform.

Parameter	Value	Default
{=}	{= | are | is}	–
value	integer	3

Dd-Ilu Preconditioner

Scope

Bicgstab Solver, Cg Solver, Expert Belos Solver, Gcrodr Solver, Gmres Solver, Poly Solver, Teko Subblock Preconditioner

Summary

Use a domain-decomposition ILU(k) preconditioner.

```
begin Dd-Ilu Preconditioner

  Fill Level {=} value

  Number Of Gpu Streams {=} value

  Reordering {=} {metis | off | on | rcm}

  Subdomain Overlap Level {=} value
```

```
Use Thread Parallel Implementation {=} {off | on}

end Dd-Ilu Preconditioner
```

Line Commands

Fill Level

Syntax

Fill Level *{=}* *value*

Summary

Set the level of fill, k, for ILU(k).

Parameter	Value	Default
{=}	{= | are | is}	–
value	integer	0

Number Of Gpu Streams

Syntax

Number Of Gpu Streams *{=}* *value*

Summary

DEVELOPER COMMAND: Set the number of GPU streams to use in GPU builds.

Description

On each MPI rank the local matrix will be further decomposed into this number of diagonal sub-matrices and the ILU preconditioner will be applied to each diagonal sub-matrix in parallel using GPU streams.

Parameter	Value	Default
{=}	{= | are | is}	–
value	integer	0

Reordering

Syntax

Reordering *{=}* *{metis | off | on | rcm}*

Summary

Choose the reordering method to apply to the local matrices.

Description

By default reordering is ON which corresponds to the reordering method that typically performs best. In CPU builds the default reorder method is RCM, but

in GPU builds it is METIS. METIS usually requires slightly higher iteration counts than RCM, but provides more parallelism for ILU on the GPU and usually runs faster on those machines as a result. If necessary REORDERING = RCM|METIS can be used to ensure the same reordering method is used regardless of machine type.

Parameter	Value	Default
{=}	{= | are | is}	–
value	{metis | off | on | rcm}	ON

Subdomain Overlap Level

Syntax

Subdomain Overlap Level *{=}* *value*

Summary

Set the level of subdomain overlap used in constructing the preconditioner.

Parameter	Value	Default
{=}	{= | are | is}	–
value	integer	0

Use Thread Parallel Implementation

Syntax

Use Thread Parallel Implementation *{=}* *{off | on}*

Summary

Control whether or not to use the thread parallel ILU implementation from Kokkos-Kernels.

Description

This will default to the appropriate value based on the platform Aria is built for, but in rare cases it may be helpful to disable the thread parallel implementation in a GPU build for debugging purposes.

Parameter	Value	Default
{=}	{= | are | is}	–
value	{off | on}	ON

Dd-Ilut Preconditioner

Scope

Bicgstab Solver, Cg Solver, Expert Belos Solver, Gcrodr Solver, Gmres Solver, Poly Solver, Teko Subblock Preconditioner

Summary

Use a domain-decomposition ILUT preconditioner.

```
begin Dd-Ilut Preconditioner

   Drop Tolerance {=} value

   Fill Fraction {=} value

   Reordering {=} {metis | off | on | rcm}

   Subdomain Overlap Level {=} value

end Dd-Ilut Preconditioner
```

Line Commands

Drop Tolerance

Syntax

Drop Tolerance *{=}* *value*

Summary

Set the drop tolerance for ilut.

Description

Off-diagonal entries of the incomplete factorization smaller than the specified tolerance will be dropped.

Parameter	Value	Default
{=}	{= | are | is}	–
value	real	0

Fill Fraction

Syntax

Fill Fraction *{=}* *value*

Summary

Set the fill fraction for ilut.

Description

This parameter controls the number of nonzero entries that are kept in each row of the incomplete factorization. Larger values will use more memory and runtime, but may result in a more effective preconditioner.

Parameter	Value	Default
{=}	{= | are | is}	–
value	real	1

Reordering

Syntax

Reordering *{=} {metis | off | on | rcm}*

Summary

Choose the reordering method to apply to the local matrices.

Description

By default reordering is ON which corresponds to the reordering method that typically performs best. In CPU builds the default reorder method is RCM, but in GPU builds it is METIS. METIS usually requires slightly higher iteration counts than RCM, but provides more parallelism for ILU on the GPU and usually runs faster on those machines as a result. If necessary REORDERING = RCM|METIS can be used to ensure the same reordering method is used regardless of machine type.

Parameter	Value	Default
{=}	{= | are | is}	–
value	{metis | off | on | rcm}	ON

Subdomain Overlap Level

Syntax

Subdomain Overlap Level *{=} value*

Summary

Set the level of subdomain overlap used in constructing the preconditioner.

Parameter	Value	Default
{=}	{= | are | is}	–
value	integer	0

Dd-Parilut Preconditioner

Scope

Bicgstab Solver, Cg Solver, Expert Belos Solver, Gcrodr Solver, Gmres Solver, Poly Solver, Teko Subblock Preconditioner

Summary

Use a domain-decomposition ParILUT preconditioner.

Description

ParILUT is similar to ILUT, but uses an iterative approach to compute the approximate L and U factors in order to enable fine grained parallelism that can improve performance on GPUs.

```
begin Dd-Parilut Preconditioner

   Fill Fraction {=} value

   Maximum Iterations {=} value

   Reordering {=} {metis | off | on | rcm}

   Subdomain Overlap Level {=} value

end Dd-Parilut Preconditioner
```

Line Commands

Fill Fraction

Syntax

Fill Fraction *{=} value*

Summary

Set the fill level for par_ilut.

Description

This parameter controls the number of nonzero entries that are kept in each row of the incomplete factorization. Larger values will use more memory and runtime, but may result in a more effective preconditioner.

Parameter	Value	Default
{=}	{= | are | is}	–
value	real	1

Maximum Iterations

Syntax

Maximum Iterations *{=}* *value*

Summary

Set the maximum number of iterations of the par_ilut algorithm.

Parameter	Value	Default
{=}	{= | are | is}	–
value	integer	5

Reordering

Syntax

Reordering *{=}* *{metis | off | on | rcm}*

Summary

Choose the reordering method to apply to the local matrices.

Description

By default reordering is ON which corresponds to the reordering method that typically performs best. In CPU builds the default reorder method is RCM, but in GPU builds it is METIS. METIS usually requires slightly higher iteration counts than RCM, but provides more parallelism for ILU on the GPU and usually runs faster on those machines as a result. If necessary REORDERING = RCM|METIS can be used to ensure the same reordering method is used regardless of machine type.

Parameter	Value	Default
{=}	{= | are | is}	–
value	{metis | off | on | rcm}	ON

Subdomain Overlap Level

Syntax

Subdomain Overlap Level *{=}* *value*

Summary

Set the level of subdomain overlap used in constructing the preconditioner.

Parameter	Value	Default
{=}	{= | are | is}	–
value	integer	0

Fastilu Preconditioner

Scope

Bicgstab Solver, Cg Solver, Expert Belos Solver, Gcrodr Solver, Gmres Solver, Poly Solver, Teko Subblock Preconditioner

Summary

Use a domain-decomposition FastILU(k) preconditioner.

Description

FastILU is a beta preconditioner that uses an iterative approximation to the standard DD-ILU preconditioner in order to expose more parallelism for potentially better performance on GPUs.

```
begin Fastilu Preconditioner

   Damping Factor {=} value

   Fill Level {=} value

   Reordering {=} {metis | off | on | rcm}

   Subdomain Overlap Level {=} value

   Sweeps {=} value

   Triangular Solver Type {=} {fast | standard}

end Fastilu Preconditioner
```

Line Commands

Damping Factor

Syntax

Damping Factor *{=}* *value*

Summary

Set the damping factor for the FastILU algorithm.

Parameter	Value	Default
{=}	{= | are | is}	–
value	real	0.2

Fill Level

Syntax

Fill Level *{=}* *value*

Summary

Set the level of fill, k, for FastILU(k).

Parameter	Value	Default
{=}	{= | are | is}	–
value	integer	0

Reordering

Syntax

Reordering *{=}* *{metis | off | on | rcm}*

Summary

Choose the reordering method to apply to the local matrices.

Description

By default reordering is ON which corresponds to the reordering method that typically performs best. In CPU builds the default reorder method is RCM, but in GPU builds it is METIS. METIS usually requires slightly higher iteration counts than RCM, but provides more parallelism for ILU on the GPU and usually runs faster on those machines as a result. If necessary REORDERING = RCM|METIS can be used to ensure the same reordering method is used regardless of machine type.

Parameter	Value	Default
{=}	{= | are | is}	–
value	{metis | off | on | rcm}	ON

Subdomain Overlap Level

Syntax

Subdomain Overlap Level *{=}* *value*

Summary

Set the level of subdomain overlap used in constructing the preconditioner.

Parameter	Value	Default
{=}	{= | are | is}	–
value	integer	0

Sweeps

Syntax

Sweeps *{=}* *value*

Summary

Set the number of sweeps for iteratively computing the ILU approximation via the FastILU algorithm.

Parameter	Value	Default
{=}	{= | are | is}	–
value	integer	5

Triangular Solver Type

Syntax

Triangular Solver Type *{=} {fast | standard}*

Summary

Set the triangular solver type for FastILU

Parameter	Value	Default
{=}	{= | are | is}	–
value	{fast | standard}	Fast

Expert Ifpack2 Preconditioner

Scope

Bicgstab Solver, Cg Solver, Expert Belos Solver, Gcrodr Solver, Gmres Solver, Poly Solver, Teko Subblock Preconditioner

Summary

Use an Ifpack2 preconditioner with parameters specified in an XML file.

```
begin Expert Ifpack2 Preconditioner

   Xml File {=} value

end Expert Ifpack2 Preconditioner
```

Line Commands

Xml File

Syntax

Xml File *{=} value*

Summary

Supply the name of an XML file containing an Ifpack2 parameter list.

Description

The supplied XML file must contain a ParameterList with a "preconditioner type:" string parameter corresponding to the desired type of Ifpack2 preconditioner as well as any additional parameters for that preconditioner.

Parameter	Value	Default
{=}	{= | are | is}	–
value	string	–

Preset Preconditioner

Scope

Bicgstab Solver, Cg Solver, Expert Belos Solver, Gcrodr Solver, Gmres Solver, Poly Solver, Teko Subblock Preconditioner

Summary

Use a preset preconditioner for a specific linear system type.

Description

The preset preconditioner for a given linear system type is intended to provide reasonable behavior for many problems of that type. If this proves to be insufficient, the user must provide more fine-grained control over preconditioner settings. The equivalent command which produces the preconditioner will be output to the log file, and this may be used as a starting point from which changes for robustness may be made (i.e. increasing Chebyshev degree in a MueLu preconditioner).

```
begin Preset Preconditioner

  Preconditioner Type {=} {continuity | displacement_multigrid | high_
↪aspect_continuity | multiphysics | scalar_transport | teko_multiphysics␣
↪| teko_ns | thermal | thermal_multigrid}

end Preset Preconditioner
```

Line Commands

Preconditioner Type

Syntax

Preconditioner Type *{=} {continuity | displacement_multigrid | high_aspect_continuity | multiphysics | scalar_transport | teko_multiphysics | teko_ns | thermal | thermal_multigrid}*

Summary

Set the preset preconditioner type

Parameter	Value	Default
{=}	{= | are | is}	–
value	{continuity | displacement_multigrid | high_aspect_continuity | multiphysics | scalar_transport | teko_multiphysics | teko_ns | thermal | thermal_multigrid}	–

Preset Preconditioner

Scope

Teko Block Preconditioner

Summary

Use a preset preconditioner for a specific linear system type.

Description

The preset preconditioner for a given linear system type is intended to provide reasonable behavior for many problems of that type. If this proves to be insufficient, the user must provide more fine-grained control over preconditioner settings. The equivalent command which produces the preconditioner will be output to the log file, and this may be used as a starting point from which changes for robustness may be made (i.e. increasing Chebyshev degree in a MueLu preconditioner).

```
begin Preset Preconditioner

  Preconditioner Type {=} {continuity | displacement_multigrid | high_
↪aspect_continuity | multiphysics | scalar_transport | teko_multiphysics␣
↪| teko_ns | thermal | thermal_multigrid}

end Preset Preconditioner
```

Line Commands

Preconditioner Type

Syntax

Preconditioner Type *{=} {continuity | displacement_multigrid | high_aspect_continuity | multiphysics | scalar_transport | teko_multiphysics | teko_ns | thermal | thermal_multigrid}*

Summary

Set the preset preconditioner type

Parameter	Value	Default
{=}	{= | are | is}	–
value	{continuity | displacement_multigrid | high_aspect_continuity | multiphysics | scalar_transport | teko_multiphysics | teko_ns | thermal | thermal_multigrid}	–

Block Gauss-Seidel Solver

Scope

Teko Preconditioner

Summary

Use a Block Gauss-Seidel solver inside a Teko preconditioner.

```
begin Block Gauss-Seidel Solver

   Use Default Inverse subblock inverse For Subblocks

   Use Inverse subblock inverse For Subblock number

   Use Upper Triangle {=} {false | true}

end Block Gauss-Seidel Solver
```

Line Commands

Use Default Inverse

Syntax

Use Default Inverse *subblock inverse* For Subblocks

Summary

Define a default solver/preconditioner to be used for across subblocks.

Parameter	Value	Default
subblock inverse	string	–

Use Inverse

Syntax

Use Inverse *subblock inverse* For Subblock *number*

Summary

Define the solver/preconditioner to be used for a given subblock.

Parameter	Value	Default
subblock inverse	string	–
number	integer	–

Use Upper Triangle

Syntax

Use Upper Triangle *{=} {false | true}*

Summary

Define whether to use upper triangular block Gauss-Seidel

Parameter	Value	Default
{=}	{= | are | is}	–
value	{false | true}	FALSE

Block Jacobi Solver

Scope

Teko Preconditioner

Summary

Use a Block Jacobi solver inside a Teko preconditioner.

```
begin Block Jacobi Solver

   Use Default Inverse subblock inverse For Subblocks

   Use Inverse subblock inverse For Subblock number

end Block Jacobi Solver
```

Line Commands

Use Default Inverse

Syntax

Use Default Inverse *subblock inverse* For Subblocks

Summary

Define a default solver/preconditioner to be used for across subblocks.

Parameter	Value	Default
subblock inverse	string	–

Use Inverse

Syntax

Use Inverse *subblock inverse* For Subblock *number*

Summary

Define the solver/preconditioner to be used for a given subblock.

Parameter	Value	Default
subblock inverse	string	–
number	integer	–

Simple Solver

Scope

Teko Preconditioner

Summary

Use a SIMPLE/SIMPLEC solver inside a Teko preconditioner.

```
begin Simple Solver

   Alpha {=} value

   Explicit Velocity Inverse Type {=} {absrowsum | diagonal | lumped}

   Use Default Inverse subblock inverse For Subblocks

   Use Inverse subblock inverse For Subblock number

end Simple Solver
```

Line Commands

Alpha

Syntax

Alpha *{=}* *value*

Summary

Set alpha value in SIMPLE block decomposition.

Parameter	Value	Default
{=}	{= | are | is}	–
value	real	1

Explicit Velocity Inverse Type

Syntax

Explicit Velocity Inverse Type *{=}* *{absrowsum | diagonal | lumped}*

Summary

Set inverse type used for velocity term when forming the Pressure Schur complement operator.

Parameter	Value	Default
{=}	{= | are | is}	–
value	{absrowsum | diagonal | lumped}	ABSROWSUM

Use Default Inverse

Syntax

Use Default Inverse *subblock inverse* For Subblocks

Summary

Define a default solver/preconditioner to be used for across subblocks.

Parameter	Value	Default
subblock inverse	string	–

Use Inverse

Syntax

Use Inverse *subblock inverse* For Subblock *number*

Summary

Define the solver/preconditioner to be used for a given subblock.

Parameter	Value	Default
subblock inverse	string	–
number	integer	–

7.6 Material Properties

An alphabetized list of material properties is provided below

A

Absorption Coefficient

Constant

Syntax

Absorption Coefficient [*{of} SpeciesName*] = Constant *{k | value}* = *k*

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
k	real	–

Copied

Syntax

Absorption Coefficient [*{of} SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	"string"	–

Exponential

Syntax

Absorption Coefficient [*{of}* *SpeciesName*] = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Scope

Aria Material

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Absorption Coefficient [*{of}* *SpeciesName*] = Global (Global_Name = *global_name* | [Component = *component*])

Scope

Aria Material

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
global_name	“string”	–
component	integer	0

Nd_Table

Syntax

Absorption Coefficient [*{of} SpeciesName*] = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Absorption Coefficient [*{of} SpeciesName*] = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Porous_Phase_Average

Syntax

Absorption Coefficient [*{of}* *SpeciesName*] = Porous_Phase_Average ([Liquid_Phase_Name = *Liquid_Phase_Name*] | [Gas_Phase_Name = *Gas_Phase_Name*])

Scope

Aria Material

Summary

Absorption coefficient for a multiphase material of two or three phases

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Liquid_Phase_Name	"string"	–
Gas_Phase_Name	"string"	–

Scalar_String_Function

Syntax

Absorption Coefficient [*{of}* *SpeciesName*] = Scalar_String_Function F = *f*

Scope

Aria Material

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	"string"	–

User_Field

Syntax

Absorption Coefficient [*{of}* *SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Absorption Coefficient [*{of}* *SpeciesName*] = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Scope

Aria Material

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1
toggle	"string"	–

User_Plugin

Syntax

Absorption Coefficient [*{of}* *SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	“string”	–
plugin_parameters	“string”	–

Absorption Cross Section

Calore_User_Sub

Syntax

Absorption Cross Section [*{of} SpeciesName*] = Calore_User_Sub (Name = *name* | Type = *type* | [Multiplier = *multiplier*] | [Material_Data_Block = *material_data_block*] | [Data = *data*] | [Scaling_Field = *scaling_field*])

Scope

Aria Material

Summary

Calore user subroutine absorption cross section

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
type	“string”	–
multiplier	real	1
material_data_block	“string”	–
data	“string”	–
scaling_field	“string”	–

Constant

Syntax

Absorption Cross Section [*{of} SpeciesName*] = Constant *{abs | value}* = *abs*

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
abs	real	–

Copied

Syntax

Absorption Cross Section [*{of}* *SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	“string”	–

Exponential

Syntax

Absorption Cross Section [*{of}* *SpeciesName*] = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Scope

Aria Material

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Absorption Cross Section [*{of}* *SpeciesName*] = Global (Global_Name = *global_name* | [Component = *component*])

Scope

Aria Material

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
global_name	"string"	–
component	integer	0

Linearized

Syntax

Absorption Cross Section [*{of}* *SpeciesName*] = Linearized (Sigma_0 = *sigma_0* | D_Sigma_D_Rho = *d_sigma_d_rho* | D_Sigma_D_T = *d_sigma_d_t* | T_Sigma_0 = *t_sigma_0* | Rho_Sigma_0 = *rho_sigma_0*)

Scope

Aria Material

Summary

Linearized absorption cross section

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
sigma_0	real	–
d_sigma_d_rho	real	–
d_sigma_d_t	real	–
t_sigma_0	real	–
rho_sigma_0	real	–

Nd_Table

Syntax

Absorption Cross Section [*{of} SpeciesName*] = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Absorption Cross Section [*{of}* *SpeciesName*] = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Absorption Cross Section [*{of}* *SpeciesName*] = Scalar_String_Function F = *f*

Scope

Aria Material

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be

a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	"string"	–

User_Field

Syntax

Absorption Cross Section [*{of}* *SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Absorption Cross Section [*{of}* *SpeciesName*] = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Scope

Aria Material

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1
toggle	"string"	–

User_Plugin

Syntax

Absorption Cross Section [*{of} SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	"string"	–
plugin_parameters	"string"	–

Activate

Binary_State

Syntax

Activate [*{of} SpeciesName*] = Binary_State Criteria = *criteria*

Scope

Aria Material

Summary

Binary State activation

Description

Provides an expression that represents the active/inactive elements with criteria for activation. Parameter CRITERIA is a string function criteria to mark an element as activated.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
criteria	"string"	–

Advected Bubble

Constant

Syntax

Advected Bubble [*{of}* *SpeciesName*] = Constant Value = *value*

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
value	real	–

Copied

Syntax

Advected Bubble [*{of}* *SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	"string"	–

Exponential

Syntax

Advected Bubble [*{of}* *SpeciesName*] = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Scope

Aria Material

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Advected Bubble [*{of}* *SpeciesName*] = Global (Global_Name = *global_name* | [Component = *component*])

Scope

Aria Material

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
global_name	“string”	–
component	integer	0

Nd_Table

Syntax

Advected Bubble [*{of}* *SpeciesName*] = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Advected Bubble [*{of}* *SpeciesName*] = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Advected Bubble [*{of} SpeciesName*] = Scalar_String_Function F = *f*

Scope

Aria Material

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	"string"	–

User_Field

Syntax

Advected Bubble [*{of}* *SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Advected Bubble [*{of}* *SpeciesName*] = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Scope

Aria Material

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Advected Bubble [*{of}* *SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	“string”	–
plugin_parameters	“string”	–

Advection Velocity

Bar

Syntax

Advection Velocity [*{of}* *SpeciesName*] = Bar ()

Scope

Aria Material

Summary

Advective bar velocity

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–

Constant

Syntax

Advection Velocity [*{of}* *SpeciesName*] = Constant ([Valuex = *valuex*] | [Valuey = *valuey*] | [Valuez = *valuez*])

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
valuex	real	0
valuey	real	0
valuez	real	0

Copied

Syntax

Advection Velocity [*{of}* *SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	“string”	–

Scalar_Vector_Product

Syntax

Advection Velocity [*{of}* *SpeciesName*] = Scalar_Vector_Product (Vector = *vector* | [Scalars = *scalars*] | [Multiplier = *multiplier*])

Scope

Aria Material

Summary

Scalar vector product

Description

Multiplies a vector by a list of provided scalars (variables) and a multiplier (constant)

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
vector	"string"	–
scalars	"string"	–
multiplier	real	1

Tensor_Vector_Product

Syntax

Advection Velocity [*{of}* *SpeciesName*] = Tensor_Vector_Product (Vector = *vector* | Tensor = *tensor* | [Scale = *scale*])

Scope

Aria Material

Summary

Tensor vector product

Description

Calculates a vector resulting from a tensor-vector product with an optional scaling factor.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
vector	"string"	–
tensor	"string"	–
scale	real	1

User_Field

Syntax

Advection Velocity [*{of} SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Vector from User Field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Advection Velocity [*{of} SpeciesName*] = User_Function ([Name_X = *name_x*] | [Name_Y = *name_y*] | [Name_Z = *name_z*] | X = *x* | [Multiplier = *multiplier*])

Scope

Aria Material

Summary

Value from a Vector User Function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name_x	"string"	–
name_y	"string"	–
name_z	"string"	–
x	"string"	–
multiplier	real	1

User_Plugin

Syntax

Advection Velocity [*{of} SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	"string"	–
plugin_parameters	"string"	–

Vector_String_Function

Syntax

Advection Velocity [*{of} SpeciesName*] = Vector_String_Function (F_X = *f_x* | F_Y = *f_y* | [F_Z = *f_z*])

Scope

Aria Material

Summary

Value from a string function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f_x	"string"	–
f_y	"string"	–
f_z	"string"	–

Aero Specific Heat Hat

Constant

Syntax

Aero Specific Heat Hat [*{of}* *SpeciesName*] = Constant Value = *value*

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
value	real	–

Copied

Syntax

Aero Specific Heat Hat [*{of}* *SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	“string”	–

Exponential

Syntax

Aero Specific Heat Hat [*{of}* *SpeciesName*] = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Scope

Aria Material

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Aero Specific Heat Hat [*{of} SpeciesName*] = Global (Global_Name = *global_name* | [Component = *component*])

Scope

Aria Material

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Aero Specific Heat Hat [*{of} SpeciesName*] = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Aero Specific Heat Hat [*{of} SpeciesName*] = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Aero Specific Heat Hat [*{of}* *SpeciesName*] = Scalar_String_Function F = *f*

Scope

Aria Material

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	"string"	–

User_Field

Syntax

Aero Specific Heat Hat [*{of}* *SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Aero Specific Heat Hat [*{of}* *SpeciesName*] = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Scope

Aria Material

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Aero Specific Heat Hat [*{of} SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	“string”	–
plugin_parameters	“string”	–

Aero Temperature Hat

Constant

Syntax

Aero Temperature Hat [*{of} SpeciesName*] = Constant Value = *value*

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
value	real	–

Copied

Syntax

Aero Temperature Hat [*{of}* *SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	“string”	–

Exponential

Syntax

Aero Temperature Hat [*{of}* *SpeciesName*] = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Scope

Aria Material

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Aero Temperature Hat [*{of} SpeciesName*] = Global (Global_Name = *global_name* | [Component = *component*])

Scope

Aria Material

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Aero Temperature Hat [*{of} SpeciesName*] = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Aero Temperature Hat [*{of}* *SpeciesName*] = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Aero Temperature Hat [*{of} SpeciesName*] = Scalar_String_Function F = *f*

Scope

Aria Material

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	"string"	–

User_Field

Syntax

Aero Temperature Hat [*{of} SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Aero Temperature Hat [*{of} SpeciesName*] = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Scope

Aria Material

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Aero Temperature Hat [*{of} SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	“string”	–
plugin_parameters	“string”	–

Aero Thermal Conductivity Hat

Constant

Syntax

Aero Thermal Conductivity Hat [*{of} SpeciesName*] = Constant Value = *value*

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
value	real	–

Copied

Syntax

Aero Thermal Conductivity Hat [*{of}* *SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	“string”	–

Exponential

Syntax

Aero Thermal Conductivity Hat [*{of}* *SpeciesName*] = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Scope

Aria Material

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Aero Thermal Conductivity Hat [*{of}* *SpeciesName*] = Global (Global_Name = *global_name* | [Component = *component*])

Scope

Aria Material

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Aero Thermal Conductivity Hat [*{of}* *SpeciesName*] = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Aero Thermal Conductivity Hat [*{of} SpeciesName*] = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Aero Thermal Conductivity Hat [*{of}* *SpeciesName*] = Scalar_String_Function F = *f*

Scope

Aria Material

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	"string"	–

User_Field

Syntax

Aero Thermal Conductivity Hat [*{of} SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Aero Thermal Conductivity Hat [*{of} SpeciesName*] = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Scope

Aria Material

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Aero Thermal Conductivity Hat [*{of}* *SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	“string”	–
plugin_parameters	“string”	–

Aero Viscosity Hat

Constant

Syntax

Aero Viscosity Hat [*{of}* *SpeciesName*] = Constant Value = *value*

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
value	real	–

Copied

Syntax

Aero Viscosity Hat [*{of}* *SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	“string”	–

Exponential

Syntax

Aero Viscosity Hat [*{of}* *SpeciesName*] = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Scope

Aria Material

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Aero Viscosity Hat [*{of} SpeciesName*] = Global (Global_Name = *global_name* | [Component = *component*])

Scope

Aria Material

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Aero Viscosity Hat [*{of} SpeciesName*] = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Aero Viscosity Hat [*{of}* *SpeciesName*] = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Aero Viscosity Hat [*{of} SpeciesName*] = Scalar_String_Function F = *f*

Scope

Aria Material

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	"string"	–

User_Field

Syntax

Aero Viscosity Hat [*{of} SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Aero Viscosity Hat [*{of} SpeciesName*] = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Scope

Aria Material

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Aero Viscosity Hat [*{of}* *SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	“string”	–
plugin_parameters	“string”	–

Air Water Mass Flux

Fickian

Syntax

Air Water Mass Flux [*{of}* *SpeciesName*] = Fickian ()

Scope

Aria Material

Summary

Fickian diffusion for the air-water mass flux

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–

Altitude

Constant

Syntax

Altitude [*{of}* *SpeciesName*] = Constant *{a | value}* = *a*

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
a	real	–

Copied

Syntax

Altitude [*{of}* *SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	“string”	–

Exponential

Syntax

Altitude [*{of}* *SpeciesName*] = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Scope

Aria Material

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Altitude [*{of} SpeciesName*] = Global (Global_Name = *global_name* | [Component = *component*])

Scope

Aria Material

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Altitude [*{of} SpeciesName*] = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Altitude [*{of} SpeciesName*] = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Altitude [*{of} SpeciesName*] = Scalar_String_Function F = *f*

Scope

Aria Material

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	"string"	–

User_Field

Syntax

Altitude [*{of}* *SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Altitude [*{of}* *SpeciesName*] = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Scope

Aria Material

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Altitude [*{of}* *SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	“string”	–
plugin_parameters	“string”	–

Ambient Pressure

Constant

Syntax

Ambient Pressure [*{of}* *SpeciesName*] = Constant Value = *value*

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
value	real	–

Copied

Syntax

Ambient Pressure [*{of}* *SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	“string”	–

Exponential

Syntax

Ambient Pressure [*{of}* *SpeciesName*] = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Scope

Aria Material

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Ambient Pressure [*{of}* *SpeciesName*] = Global (Global_Name = *global_name* | [Component = *component*])

Scope

Aria Material

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Ambient Pressure [*{of}* *SpeciesName*] = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Ambient Pressure [*{of}* *SpeciesName*] = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Ambient Pressure [*{of} SpeciesName*] = Scalar_String_Function F = *f*

Scope

Aria Material

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	"string"	–

User_Field

Syntax

Ambient Pressure [*{of}* *SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Ambient Pressure [*{of}* *SpeciesName*] = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Scope

Aria Material

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Ambient Pressure [*{of}* *SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	“string”	–
plugin_parameters	“string”	–

Angular Intensity

Constant

Syntax

Angular Intensity [*{of}* *SpeciesName*] = Constant Value = *value*

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
value	real	–

Copied

Syntax

Angular Intensity [*{of}* *SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	“string”	–

Exponential

Syntax

Angular Intensity [*{of}* *SpeciesName*] = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Scope

Aria Material

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Angular Intensity [*{of}* *SpeciesName*] = Global (Global_Name = *global_name* | [Component = *component*])

Scope

Aria Material

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Angular Intensity [*{of}* *SpeciesName*] = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Angular Intensity [*{of}* *SpeciesName*] = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Angular Intensity [*{of}* *SpeciesName*] = Scalar_String_Function F = *f*

Scope

Aria Material

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	"string"	–

User_Field

Syntax

Angular Intensity [*{of}* *SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Angular Intensity [*{of}* *SpeciesName*] = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Scope

Aria Material

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Angular Intensity [*{of} SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	“string”	–
plugin_parameters	“string”	–

Annulus Diameter Ratio

Constant

Syntax

Annulus Diameter Ratio [*{of} SpeciesName*] = Constant *{r | value}* = *r*

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
r	real	–

Copied

Syntax

Annulus Diameter Ratio [*{of}* *SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	“string”	–

Correlation

Syntax

Annulus Diameter Ratio [*{of}* *SpeciesName*] = Correlation ([R = *r*] | [Threshold = *threshold*] | [Bulk_Node_Interface = *bulk_node_interface*])

Scope

Aria Material

Summary

Correlation annulus diameter ratio based on hydraulic diameter and wetted perimeter

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
r	real	0
threshold	real	0.05
bulk_node_interface	“string”	–

Correlation_Spatial_User_Function

Syntax

Annulus Diameter Ratio [*{of}* *SpeciesName*] = Correlation_Spatial_User_Function ([Threshold = *threshold*] | [Multiplier = *multiplier*] | Name = *name* | Bar = *bar* | X = *x* | [Bulk_Node_Interface = *bulk_node_interface*])

Scope

Aria Material

Summary

Correlation spatially varying annulus diameter ratio based on a coordinate dependent user function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
threshold	real	-1.798e+308
multiplier	real	1
name	"string"	–
bar	"string"	–
x	"string"	–
bulk_node_interface	"string"	–

Exponential

Syntax

Annulus Diameter Ratio [*{of}* *SpeciesName*] = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Scope

Aria Material

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Annulus Diameter Ratio [*{of} SpeciesName*] = Global (Global_Name = *global_name* | [Component = *component*])

Scope

Aria Material

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Annulus Diameter Ratio [*{of} SpeciesName*] = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Annulus Diameter Ratio [*{of}* *SpeciesName*] = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Annulus Diameter Ratio [*{of}* *SpeciesName*] = Scalar_String_Function F = *f*

Scope

Aria Material

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	"string"	–

Spatial_User_Function

Syntax

Annulus Diameter Ratio [*{of} SpeciesName*] = Spatial_User_Function ([Threshold = *threshold*] | [Multiplier = *multiplier*] | Name = *name* | Bar = *bar* | X = *x*)

Scope

Aria Material

Summary

Spatially varying annulus diameter ratio based on a coordinate dependent user function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
threshold	real	-1.798e+308
multiplier	real	1
name	“string”	–
bar	“string”	–
x	“string”	–

User_Field

Syntax

Annulus Diameter Ratio [*{of} SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
scaling	real	1
global_var	“string”	–
time_function	“string”	–

User_Function

Syntax

Annulus Diameter Ratio [*{of}* *SpeciesName*] = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Scope

Aria Material

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1
toggle	"string"	–

User_Plugin

Syntax

Annulus Diameter Ratio [*{of}* *SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	“string”	–
plugin_parameters	“string”	–

Auxiliary Mesh Field

Scalar

Syntax

Auxiliary Mesh Field [*{of} SpeciesName*] = Scalar (Field_Name = *field_name* | [Expression_Name = *expression_name*])

Scope

Aria Material

Summary

Scalar auxiliary mesh field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
field_name	“string”	–
expression_name	“string”	–

B

Bar Area

Constant

Syntax

Bar Area [*{of} SpeciesName*] = Constant *{a | value}* = *a*

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
a	real	–

Copied

Syntax

Bar Area [*{of}* *SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	“string”	–

Element_Attribute

Syntax

Bar Area [*{of}* *SpeciesName*] = Element_Attribute ([Multiplier = *multiplier*] | [Attribute_Name = *attribute_name*])

Scope

Aria Material

Summary

Distribution Factor for the bar area

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
multiplier	real	1
attribute_name	“string”	DEFAULT_AREA

Exponential

Syntax

Bar Area [*{of} SpeciesName*] = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Scope

Aria Material

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bar Area [*{of} SpeciesName*] = Global (Global_Name = *global_name* | [Component = *component*])

Scope

Aria Material

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Bar Area [*{of} SpeciesName*] = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bar Area [*{of} SpeciesName*] = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bar Area [*{of}* *SpeciesName*] = Scalar_String_Function F = *f*

Scope

Aria Material

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	"string"	–

User_Field

Syntax

Bar Area [*{of}* *SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bar Area [*{of}* *SpeciesName*] = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Scope

Aria Material

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1
toggle	"string"	–

User_Plugin

Syntax

Bar Area [*{of}* *SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	"string"	–
plugin_parameters	"string"	–

Bar Flux Contribution Factor

Constant

Syntax

Bar Flux Contribution Factor [*{of}* *SpeciesName*] = Constant Value = *value*

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
value	real	–

Copied

Syntax

Bar Flux Contribution Factor [*{of}* *SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	“string”	–

Exponential

Syntax

Bar Flux Contribution Factor [*{of}* *SpeciesName*] = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Scope

Aria Material

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bar Flux Contribution Factor [*{of} SpeciesName*] = Global (Global_Name = *global_name* | [Component = *component*])

Scope

Aria Material

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Bar Flux Contribution Factor [*{of} SpeciesName*] = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bar Flux Contribution Factor [*{of}* *SpeciesName*] = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bar Flux Contribution Factor [*{of} SpeciesName*] = Scalar_String_Function F = *f*

Scope

Aria Material

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	"string"	–

User_Field

Syntax

Bar Flux Contribution Factor [*{of}* *SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bar Flux Contribution Factor [*{of}* *SpeciesName*] = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Scope

Aria Material

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Bar Flux Contribution Factor [*{of} SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	“string”	–
plugin_parameters	“string”	–

Bar Perimeter

Constant

Syntax

Bar Perimeter [*{of} SpeciesName*] = Constant *{p | value}* = *p*

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
p	real	–

Copied

Syntax

Bar Perimeter [*{of}* *SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	“string”	–

Element_Attribute

Syntax

Bar Perimeter [*{of}* *SpeciesName*] = Element_Attribute ([Multiplier = *multiplier*] | [Attribute_Name = *attribute_name*])

Scope

Aria Material

Summary

Distribution Factor for the bar perimeter

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
multiplier	real	1
attribute_name	“string”	DEFAULT_PERIMETER

Exponential

Syntax

Bar Perimeter [*{of}* *SpeciesName*] = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Scope

Aria Material

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bar Perimeter [*{of}* *SpeciesName*] = Global (Global_Name = *global_name* | [Component = *component*])

Scope

Aria Material

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Bar Perimeter [*{of}* *SpeciesName*] = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bar Perimeter [*{of}* *SpeciesName*] = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bar Perimeter [*{of}* *SpeciesName*] = Scalar_String_Function F = *f*

Scope

Aria Material

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	"string"	–

User_Field

Syntax

Bar Perimeter [*{of}* *SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bar Perimeter [*{of}* *SpeciesName*] = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Scope

Aria Material

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Bar Perimeter [*{of} SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	“string”	–
plugin_parameters	“string”	–

Bc Environment Temperature

Constant

Syntax

Bc Environment Temperature [*{of} SpeciesName*] = Constant *{t | value}* = *t*

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
t	real	–

Copied

Syntax

Bc Environment Temperature [*{of}* *SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	“string”	–

Exponential

Syntax

Bc Environment Temperature [*{of}* *SpeciesName*] = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Scope

Aria Material

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Environment Temperature [*{of}* *SpeciesName*] = Global (Global_Name = *global_name* | [Component = *component*])

Scope

Aria Material

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Bc Environment Temperature [*{of}* *SpeciesName*] = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc Environment Temperature [*{of}* *SpeciesName*] = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Environment Temperature [*{of}* *SpeciesName*] = Scalar_String_Function F = *f*

Scope

Aria Material

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	"string"	–

User_Field

Syntax

Bc Environment Temperature [*{of} SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc Environment Temperature [*{of} SpeciesName*] = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Scope

Aria Material

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1
toggle	"string"	–

User_Plugin

Syntax

Bc Environment Temperature [*{of}* *SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	"string"	–
plugin_parameters	"string"	–

Bc Heat Flux

Constant

Syntax

Bc Heat Flux [*{of}* *SpeciesName*] = Constant ([Qx = *qX*] | [Qy = *qY*] | [Qz = *qZ*])

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
qX	real	0
qY	real	0
qZ	real	0

Copied

Syntax

Bc Heat Flux [*{of}* *SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	“string”	–

Scalar_Vector_Product

Syntax

Bc Heat Flux [*{of}* *SpeciesName*] = Scalar_Vector_Product (Vector = *vector* | [Scalars = *scalars*] | [Multiplier = *multiplier*])

Scope

Aria Material

Summary

Scalar vector product

Description

Multiplies a vector by a list of provided scalars (variables) and a multiplier (constant)

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
vector	“string”	–
scalars	“string”	–
multiplier	real	1

Tensor_Vector_Product

Syntax

Bc Heat Flux [*{of}* *SpeciesName*] = Tensor_Vector_Product (Vector = *vector* | Tensor = *tensor* | [Scale = *scale*])

Scope

Aria Material

Summary

Tensor vector product

Description

Calculates a vector resulting from a tensor-vector product with an optional scaling factor.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
vector	“string”	–
tensor	“string”	–
scale	real	1

User_Field

Syntax

Bc Heat Flux [*{of}* *SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Vector from User Field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Field_Times_Inward_Normal

Syntax

Bc Heat Flux [*{of}* *SpeciesName*] = User_Field_Times_Inward_Normal (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Transform a scalar flux term into a vector quantity by multiplying by the inward surface normal.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc Heat Flux [*{of}* *SpeciesName*] = User_Function ([Name_X = *name_x*] | [Name_Y = *name_y*] | [Name_Z = *name_z*] | X = *x* | [Multiplier = *multiplier*])

Scope

Aria Material

Summary

Value from a Vector User Function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name_x	"string"	–
name_y	"string"	–
name_z	"string"	–
x	"string"	–
multiplier	real	1

User_Plugin

Syntax

Bc Heat Flux [*{of}* *SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	"string"	–
plugin_parameters	"string"	–

Vector_String_Function

Syntax

Bc Heat Flux [*{of}* *SpeciesName*] = Vector_String_Function (F_X = *f_x* | F_Y = *f_y* | [F_Z = *f_z*])

Scope

Aria Material

Summary

Value from a string function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f_x	“string”	–
f_y	“string”	–
f_z	“string”	–

Bc Rad Reference Temperature

Bulk_Node

Syntax

Bc Rad Reference Temperature [*{of} SpeciesName*] = Bulk_Node Name = *name*

Scope

Aria Material

Summary

Use a bulk node temperature as the reference temperature for radiation boundary conditions

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–

Calore_User_Sub

Syntax

Bc Rad Reference Temperature [*{of} SpeciesName*] = Calore_User_Sub (Name = *name* | Type = *type* | [Multiplier = *multiplier*] | [Material_Data_Block = *material_data_block*] | [Data = *data*] | [Scaling_Field = *scaling_field*])

Scope

Aria Material

Summary

Values from a Calore user subroutine

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
type	“string”	–
multiplier	real	1
material_data_block	“string”	–
data	“string”	–
scaling_field	“string”	–

Constant

Syntax

Bc Rad Reference Temperature [*{of}* *SpeciesName*] = Constant *{t_ref | value}* = *t_ref*

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
t_ref	real	–

Copied

Syntax

Bc Rad Reference Temperature [*{of}* *SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	“string”	–

Exponential

Syntax

Bc Rad Reference Temperature [*{of}* *SpeciesName*] = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Scope

Aria Material

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Fortran

Syntax

Bc Rad Reference Temperature [*{of}* *SpeciesName*] = Fortran ([Multiplier = *multiplier*] | Sub_Name = *sub_name* | [Real_Data = *real_data*] | [Int_Data = *int_data*] | [Resource_Name = *resource_name*] | [Data = *data*])

Scope

Aria Material

Summary

Fortran subroutine reference temperature.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
multiplier	real	1
sub_name	“string”	–
real_data	“string”	–
int_data	“string”	–
resource_name	“string”	–
data	“string”	–

Global

Syntax

Bc Rad Reference Temperature [*{of} SpeciesName*] = Global (Global_Name = *global_name* | [Component = *component*])

Scope

Aria Material

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
global_name	“string”	–
component	integer	0

Nd_Table

Syntax

Bc Rad Reference Temperature [*{of} SpeciesName*] = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc Rad Reference Temperature [*{of}* *SpeciesName*] = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Rad Reference Temperature [*{of} SpeciesName*] = Scalar_String_Function F = *f*

Scope

Aria Material

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	"string"	–

User_Field

Syntax

Bc Rad Reference Temperature [*{of} SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc Rad Reference Temperature [*{of} SpeciesName*] = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Scope

Aria Material

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Bc Rad Reference Temperature [*{of} SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	“string”	–
plugin_parameters	“string”	–

Bc Reference Temperature

Bulk_Node

Syntax

Bc Reference Temperature [*{of} SpeciesName*] = Bulk_Node Name = *name*

Scope

Aria Material

Summary

Use a bulk node temperature as the reference temperature for convection boundary conditions

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–

Calore_User_Sub

Syntax

Bc Reference Temperature [*{of}* *SpeciesName*] = Calore_User_Sub (Name = *name* | Type = *type* | [Multiplier = *multiplier*] | [Material_Data_Block = *material_data_block*] | [Data = *data*] | [Scaling_Field = *scaling_field*])

Scope

Aria Material

Summary

Values from a Calore user subroutine

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
type	“string”	–
multiplier	real	1
material_data_block	“string”	–
data	“string”	–
scaling_field	“string”	–

Computed_Adiabatic_Wall

Syntax

Bc Reference Temperature [*{of}* *SpeciesName*] = Computed_Adiabatic_Wall (T_On = *t_on* | T_Off = *t_off* | Offset_Direction = *offset_direction* | Coord_Offset = *coord_offset* | Gas_Constant = *gas_constant*)

Scope

Aria Material

Summary

Adiabatic wall temperature computed based upon freestream and recovery factor.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
t_on	real	–
t_off	real	–
offset_direction	“string”	–
coord_offset	real	–
gas_constant	real	–

Constant

Syntax

Bc Reference Temperature [*{of} SpeciesName*] = Constant *{t_ref | value}* = *t_ref*

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
t_ref	real	–

Copied

Syntax

Bc Reference Temperature [*{of} SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	“string”	–

Exponential

Syntax

Bc Reference Temperature [*{of}* *SpeciesName*] = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Scope

Aria Material

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Fortran

Syntax

Bc Reference Temperature [*{of}* *SpeciesName*] = Fortran ([Multiplier = *multiplier*] | Sub_Name = *sub_name* | [Real_Data = *real_data*] | [Int_Data = *int_data*] | [Resource_Name = *resource_name*] | [Data = *data*])

Scope

Aria Material

Summary

Fortran subroutine reference temperature.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
multiplier	real	1
sub_name	“string”	–
real_data	“string”	–
int_data	“string”	–
resource_name	“string”	–
data	“string”	–

Global

Syntax

Bc Reference Temperature [*{of}* *SpeciesName*] = Global (Global_Name = *global_name* | [Component = *component*])

Scope

Aria Material

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
global_name	“string”	–
component	integer	0

Nd_Table

Syntax

Bc Reference Temperature [*{of}* *SpeciesName*] = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc Reference Temperature [*{of}* *SpeciesName*] = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Reference Temperature [*{of} SpeciesName*] = Scalar_String_Function F = *f*

Scope

Aria Material

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	"string"	–

User_Field

Syntax

Bc Reference Temperature [*{of}* *SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc Reference Temperature [*{of}* *SpeciesName*] = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Scope

Aria Material

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Bc Reference Temperature [*{of} SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	“string”	–
plugin_parameters	“string”	–

Beta

Constant

Syntax

Beta [*{of} SpeciesName*] = Constant *{beta | value}* = *beta*

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
beta	real	–

Converted

Syntax

Beta [*{of}* *SpeciesName*] = Converted ()

Scope

Aria Material

Summary

Coefficient of thermal stress computed from Poisson's ratio, thermal expansion coefficient and elastic modulus

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–

Copied

Syntax

Beta [*{of}* *SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	"string"	–

Exponential

Syntax

Beta [*{of}* *SpeciesName*] = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Scope

Aria Material

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Beta [*{of}* *SpeciesName*] = Global (Global_Name = *global_name* | [Component = *component*])

Scope

Aria Material

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
global_name	"string"	–
component	integer	0

Linear

Syntax

Beta [*{of} SpeciesName*] = Linear (A = *a* | B = *b*)

Scope

Aria Material

Summary

This model has been deprecated and removed from the code.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
a	real	–
b	real	–

Nd_Table

Syntax

Beta [*{of} SpeciesName*] = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Beta [*{of}* *SpeciesName*] = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Beta [*{of}* *SpeciesName*] = Scalar_String_Function F = *f*

Scope

Aria Material

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be

a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	"string"	–

User_Field

Syntax

Beta [*{of}* *SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Beta [*{of}* *SpeciesName*] = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Scope

Aria Material

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1
toggle	"string"	–

User_Plugin

Syntax

Beta [*{of}* *SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	"string"	–
plugin_parameters	"string"	–

Body Acceleration

Constant

Syntax

Body Acceleration [*{of}* *SpeciesName*] = Constant ([Valuex = *valuex*] | [Valuey = *valuey*] | [Valuez = *valuez*])

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
valuex	real	0
valuey	real	0
valuez	real	0

Copied

Syntax

Body Acceleration [*{of}* *SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	“string”	–

Scalar_Vector_Product

Syntax

Body Acceleration [*{of}* *SpeciesName*] = Scalar_Vector_Product (Vector = *vector* | [Scalars = *scalars*] | [Multiplier = *multiplier*])

Scope

Aria Material

Summary

Scalar vector product

Description

Multiplies a vector by a list of provided scalars (variables) and a multiplier (constant)

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
vector	"string"	–
scalars	"string"	–
multiplier	real	1

Tensor_Vector_Product

Syntax

Body Acceleration [*{of}* *SpeciesName*] = Tensor_Vector_Product (Vector = *vector* | Tensor = *tensor* | [Scale = *scale*])

Scope

Aria Material

Summary

Tensor vector product

Description

Calculates a vector resulting from a tensor-vector product with an optional scaling factor.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
vector	"string"	–
tensor	"string"	–
scale	real	1

User_Field

Syntax

Body Acceleration [*{of}* *SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Vector from User Field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Body Acceleration [*{of}* *SpeciesName*] = User_Function ([Name_X = *name_x*] | [Name_Y = *name_y*] | [Name_Z = *name_z*] | X = *x* | [Multiplier = *multiplier*])

Scope

Aria Material

Summary

Value from a Vector User Function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name_x	"string"	–
name_y	"string"	–
name_z	"string"	–
x	"string"	–
multiplier	real	1

User_Plugin

Syntax

Body Acceleration [*{of}* *SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	"string"	–
plugin_parameters	"string"	–

Vector_String_Function

Syntax

Body Acceleration [*{of}* *SpeciesName*] = Vector_String_Function (F_X = *f_x* | F_Y = *f_y* | [F_Z = *f_z*])

Scope

Aria Material

Summary

Value from a string function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f_x	"string"	–
f_y	"string"	–
f_z	"string"	–

Boundary Entrained Enthalpy

Constant

Syntax

Boundary Entrained Enthalpy [*{of}* *SpeciesName*] = Constant *{h | value}* = *h*

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
h	real	–

Copied

Syntax

Boundary Entrained Enthalpy [*{of}* *SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	"string"	–

Exponential

Syntax

Boundary Entrained Enthalpy [*{of}* *SpeciesName*] = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Scope

Aria Material

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Boundary Entrained Enthalpy [*{of}* *SpeciesName*] = Global (Global_Name = *global_name* | [Component = *component*])

Scope

Aria Material

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
global_name	“string”	–
component	integer	0

Nd_Table

Syntax

Boundary Entrained Enthalpy [*{of}* *SpeciesName*] = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Boundary Entrained Enthalpy [*{of}* *SpeciesName*] = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Boundary Entrained Enthalpy [*{of}* *SpeciesName*] = Scalar_String_Function F = *f*

Scope

Aria Material

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	"string"	–

User_Field

Syntax

Boundary Entrained Enthalpy [*{of}* *SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Boundary Entrained Enthalpy [*{of}* *SpeciesName*] = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Scope

Aria Material

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1
toggle	"string"	–

User_Plugin

Syntax

Boundary Entrained Enthalpy [*{of} SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	"string"	–
plugin_parameters	"string"	–

Boundary Entrained Mass Fraction

Constant

Syntax

Boundary Entrained Mass Fraction [*{of} SpeciesName*] = Constant *{value | y}* = *y*

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
y	real	–

Copied

Syntax

Boundary Entrained Mass Fraction [*{of}* *SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	“string”	–

Exponential

Syntax

Boundary Entrained Mass Fraction [*{of}* *SpeciesName*] = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Scope

Aria Material

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Boundary Entrained Mass Fraction [*{of} SpeciesName*] = Global (Global_Name = *global_name* | [Component = *component*])

Scope

Aria Material

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Boundary Entrained Mass Fraction [*{of} SpeciesName*] = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Boundary Entrained Mass Fraction [*{of}* *SpeciesName*] = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Boundary Entrained Mass Fraction [*{of} SpeciesName*] =
Scalar_String_Function F = *f*

Scope

Aria Material

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	"string"	–

User_Field

Syntax

Boundary Entrained Mass Fraction [*{of} SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Boundary Entrained Mass Fraction [*{of} SpeciesName*] = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Scope

Aria Material

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Boundary Entrained Mass Fraction [*{of} SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	“string”	–
plugin_parameters	“string”	–

Boundary Pressure

Constant

Syntax

Boundary Pressure [*{of} SpeciesName*] = Constant *{p | value}* = *p*

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
p	real	–

Copied

Syntax

Boundary Pressure [*{of}* *SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	“string”	–

Exponential

Syntax

Boundary Pressure [*{of}* *SpeciesName*] = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Scope

Aria Material

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Boundary Pressure [*{of} SpeciesName*] = Global (Global_Name = *global_name* | [Component = *component*])

Scope

Aria Material

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Boundary Pressure [*{of} SpeciesName*] = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Boundary Pressure [*{of}* *SpeciesName*] = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Boundary Pressure [*{of} SpeciesName*] = Scalar_String_Function F = *f*

Scope

Aria Material

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	"string"	–

User_Field

Syntax

Boundary Pressure [*{of}* *SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Boundary Pressure [*{of}* *SpeciesName*] = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Scope

Aria Material

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Boundary Pressure [*{of} SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	“string”	–
plugin_parameters	“string”	–

Bulk Conductivity

Constant

Syntax

Bulk Conductivity [*{of} SpeciesName*] = Constant *{kbulk | value}* = *Kbulk*

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Kbulk	real	–

Copied

Syntax

Bulk Conductivity [*{of}* *SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	"string"	–

Exponential

Syntax

Bulk Conductivity [*{of}* *SpeciesName*] = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Scope

Aria Material

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bulk Conductivity [*{of}* *SpeciesName*] = Global (Global_Name = *global_name* | [Component = *component*])

Scope

Aria Material

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
global_name	"string"	–
component	integer	0

Linear_Temperature_And_Density

Syntax

Bulk Conductivity [*{of}* *SpeciesName*] = Linear_Temperature_And_Density (C_0 = *c_0* | C_Rho = *C_rho* | C_T = *c_t*)

Scope

Aria Material

Summary

Bulk conductivity that is a linear function of both temperature and density. $\kappa_{bulk} = C_0 + C_\rho \rho + C_T T$

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
c_0	real	–
C_rho	real	–
c_t	real	–

Nd_Table

Syntax

Bulk Conductivity [*{of} SpeciesName*] = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bulk Conductivity [*{of}* *SpeciesName*] = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Porous_Phase_Average

Syntax

Bulk Conductivity [*{of}* *SpeciesName*] = Porous_Phase_Average ([Liquid_Phase_Name = *Liquid_Phase_Name*] | [Gas_Phase_Name = *Gas_Phase_Name*])

Scope

Aria Material

Summary

Bulk conductivity for a multiphase material of two or three phases

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Liquid_Phase_Name	"string"	–
Gas_Phase_Name	"string"	–

Scalar_String_Function

Syntax

Bulk Conductivity [*{of}* *SpeciesName*] = Scalar_String_Function F = *f*

Scope

Aria Material

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	"string"	–

T_Exponent

Syntax

Bulk Conductivity [*{of}* *SpeciesName*] = T_Exponent (Kbulk_Ref = *kbulk_ref* \| T_Ref = *t_ref* \| N = *n*)

Scope

Aria Material

Summary

Thermal conductivity for bulk material.$\kappa_{\text{bulk}} = \kappa_{\text{ref}} \frac{\rho_{\text{solid}}}{\rho_{\text{bulk}}} \left(\frac{T}{T_{\text{ref}}} \right)^n$.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
kbulk_ref	real	–
t_ref	real	–
n	real	–

User_Field

Syntax

Bulk Conductivity [*{of}* *SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bulk Conductivity [*{of}* *SpeciesName*] = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Scope

Aria Material

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1
toggle	"string"	–

User_Plugin

Syntax

Bulk Conductivity [*{of} SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	"string"	–
plugin_parameters	"string"	–

Volume_Average

Syntax

Bulk Conductivity [*{of} SpeciesName*] = Volume_Average ()

Scope

Aria Material

Summary

Bulk conductivity computed from volume average of species bulk conductivities

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–

Bulk Density

Constant

Syntax

Bulk Density [*{of}* *SpeciesName*] = Constant *{rho_b | value}* = *rho_b*

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
rho_b	real	–

Copied

Syntax

Bulk Density [*{of}* *SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	“string”	–

Exponential

Syntax

Bulk Density [*{of}* *SpeciesName*] = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Scope

Aria Material

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bulk Density [*{of}* *SpeciesName*] = Global (Global_Name = *global_name* | [Component = *component*])

Scope

Aria Material

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Bulk Density [*{of}* *SpeciesName*] = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bulk Density [*{of}* *SpeciesName*] = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bulk Density [*{of}* *SpeciesName*] = Scalar_String_Function F = *f*

Scope

Aria Material

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	"string"	–

Single_Component_Ideal_Gas

Syntax

Bulk Density [*{of}* *SpeciesName*] = Single_Component_Ideal_Gas ([P_Ref = *p_ref*] | [T_Ref = *t_ref*] | [M = *m*] | R = *r*)

Scope

Aria Material

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
p_ref	real	0
t_ref	real	0
m	real	1
r	real	–

T_Exponent

Syntax

Bulk Density [*{of}* *SpeciesName*] = T_Exponent (Rho_Ref = *rho_ref* | T_Ref = *t_ref* | N = *n*)

Scope

Aria Material

Summary

Bulk density as a power of normalized temperature$\rho_{\text{bulk}} = \rho_{\text{ref}} \left(\frac{T}{T_{\text{ref}}} \right)^n$.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
rho_ref	real	–
t_ref	real	–
n	real	–

User_Field

Syntax

Bulk Density [*{of}* *SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bulk Density [*{of}* *SpeciesName*] = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Scope

Aria Material

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Bulk Density [*{of}* *SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	“string”	–
plugin_parameters	“string”	–

Bulk Element Pressure

Calore_User_Sub

Syntax

Bulk Element Pressure [*{of}* *SpeciesName*] = Calore_User_Sub (Name = *name* | Type = *type* | [Multiplier = *multiplier*] | [Material_Data_Block = *material_data_block*] | [Data = *data*] | [Scaling_Field = *scaling_field*] | Bulk_Node = *bulk_node*)

Scope

Aria Material

Summary

Bulk element pressure from Calore User Subroutine

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
type	“string”	–
multiplier	real	1
material_data_block	“string”	–
data	“string”	–
scaling_field	“string”	–
bulk_node	“string”	–

Constant

Syntax

Bulk Element Pressure [*{of}* *SpeciesName*] = Constant V = *v*

Scope

Aria Material

Summary

Constant value for the bulk element pressure

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
v	real	–

Integrated

Syntax

Bulk Element Pressure [*{of}* *SpeciesName*] = Integrated Name = *name*

Scope

Aria Material

Summary

Bulk element pressure computed from equation of state

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–

User_Function

Syntax

Bulk Element Pressure [*{of}* *SpeciesName*] = User_Function ()

Scope

Aria Material

Summary

Bulk element pressure from User Function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–

Bulk Element Volume

Calore_User_Sub

Syntax

Bulk Element Volume [*{of}* *SpeciesName*] = Calore_User_Sub (Name = *name* | Type = *type* | [Multiplier = *multiplier*] | [Material_Data_Block = *material_data_block*] | [Data = *data*] | [Scaling_Field = *scaling_field*] | Bulk_Node = *bulk_node*)

Scope

Aria Material

Summary

Bulk element volume from Calore User Subroutine

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
type	“string”	–
multiplier	real	1
material_data_block	“string”	–
data	“string”	–
scaling_field	“string”	–
bulk_node	“string”	–

Constant

Syntax

Bulk Element Volume [*{of}* *SpeciesName*] = Constant V = *v*

Scope

Aria Material

Summary

Constant value for the bulk element volume

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
v	real	–

Integrated

Syntax

Bulk Element Volume [*{of}* *SpeciesName*] = Integrated Name = *name*

Scope

Aria Material

Summary

Bulk element volume computed from bounding surfaces

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–

User_Function

Syntax

Bulk Element Volume [*{of}* *SpeciesName*] = User_Function Bulk_Node = *bulk_node*

Scope

Aria Material

Summary

This model is now deprecated

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
bulk_node	“string”	–

Bulk Mass Density

Constant

Syntax

Bulk Mass Density [*{of}* *SpeciesName*] = Constant Value = *value*

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
value	real	–

Copied

Syntax

Bulk Mass Density [*{of}* *SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	“string”	–

Density

Syntax

Bulk Mass Density [*{of}* *SpeciesName*] = Density ()

Scope

Aria Material

Summary

Provides the mass term for the mass balance equation where bulk mass density is copied from density.

Description

For species i, this model provides the bulk mass density and its temporal derivative resulting in the mass term

$$\frac{\partial \rho_{i,s}}{\partial t}.$$

where $rho_{i,s}$ is species solid phase density. If no species is set, this is applied to the general property. This model is only applicable to the solid phase density (or no material phase), where the bulk mass density is per total volume and not per phase volume.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–

Exponential

Syntax

Bulk Mass Density [*{of}* *SpeciesName*] = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Scope

Aria Material

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bulk Mass Density [*{of}* *SpeciesName*] = Global (Global_Name = *global_name* | [Component = *component*])

Scope

Aria Material

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
global_name	"string"	–
component	integer	0

Mass_Fraction_Density

Syntax

Bulk Mass Density [*{of}* *SpeciesName*] = Mass_Fraction_Density ()

Scope

Aria Material

Summary

Provides the mass term for the mass balance equation where bulk mass density is computed from mass fraction and density.

Description

For species i, this model provides the bulk mass density and its temporal derivative resulting in the mass term

$$\frac{\partial Y_{i,s}\rho_s}{\partial t},$$

where $Y_{i,s}$ is the species mass fraction in the solid phase and ρ_s is the solid density. Note that this model is only applicable to the solid phase (or no material phase) and requires specifying the species. The bulk mass density is per total volume and not per phase volume.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–

Mass_Fraction_Porous_Density

Syntax

Bulk Mass Density [*{of}* *SpeciesName*] = Mass_Fraction_Porous_Density ()

Scope

Aria Material

Summary

Provides the mass term for the mass balance equation where the bulk mass density is computed from mass fraction, porosity, density and optionally the saturation.

Description

For species i and phase p, this model provides the bulk mass density and its temporal derivative resulting in the mass term

$$\frac{\partial Y_{i,p}\phi S_p \rho_p}{\partial t},$$

where $Y_{i,p}$ is the species mass fraction in the given phase, ϕ is the porosity, S_p is an optional phase saturation, and ρ_p is the phase density. If the optional phase saturation S_p is provided, the pore volume is assumed to be occupied by multiple phases (e.g., liquid / gas) in accordance to their saturations, otherwise it is occupied fully by a single phase. This model is not applicable to the solid phase, which defines bulk density as per total volume, not per phase volume.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–

Multiphase_Porous_Density

Syntax

Bulk Mass Density [*{of} SpeciesName*] = Multiphase_Porous_Density ()

Scope

Aria Material

Summary

Provides the mass term for the mass balance equation where bulk mass density is computed from density, saturation and porosity for a multiphase porous material.

Description

For species i and phase p, this model provides the bulk mass density and its temporal derivative resulting in the mass term

$$\frac{\partial \phi S_p \rho_{i,p}}{\partial t},$$

where ϕ is the porosity, S_p is the phase saturation, and $\rho_{i,p}$ is the species density in the given phase. If no species is set, this is applied to the general property.

Note this model assumes the pore volume is occupied by multiple phases (e.g., liquid / gas) in accordance to the specified phase saturations. The phase saturation must be provided, it is not optional. This model is not applicable to the solid phase.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–

Nd_Table

Syntax

Bulk Mass Density [*{of} SpeciesName*] = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bulk Mass Density [*{of}* *SpeciesName*] = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Porous_Density

Syntax

Bulk Mass Density [*{of}* *SpeciesName*] = Porous_Density ()

Scope

Aria Material

Summary

Provides the mass term for the mass balance equation, where the bulk mass density is computed from density, porosity, and optionally saturation.

Description

For species i and phase p, this model provides the bulk mass density and its temporal derivative resulting in the mass term

$$\frac{\partial \phi S_p \rho_{i,p}}{\partial t},$$

where ϕ is the porosity, S_p is the optional phase saturation, and $\rho_{i,p}$ is the species density in the given phase. If no species is set, this is applied to the general property.

This model is not applicable to the solid phase, which defines bulk density as mass per total volume, not per phase volume. If the optional phase saturation S_p is provided, the pore volume is assumed to be occupied by multiple phases (e.g., liquid / gas) in accordance to their saturations, otherwise it is occupied fully by a single phase.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–

Scalar_String_Function

Syntax

Bulk Mass Density [*{of}* *SpeciesName*] = Scalar_String_Function F = *f*

Scope

Aria Material

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	"string"	–

User_Field

Syntax

Bulk Mass Density [*{of}* *SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bulk Mass Density [*{of}* *SpeciesName*] = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Scope

Aria Material

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1
toggle	"string"	–

User_Plugin

Syntax

Bulk Mass Density [*{of} SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	“string”	–
plugin_parameters	“string”	–

Bulk Species

Constant

Syntax

Bulk Species [*{of} SpeciesName*] = Constant Value = *value*

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
value	real	–

Copied

Syntax

Bulk Species [*{of}* *SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	"string"	–

Exponential

Syntax

Bulk Species [*{of}* *SpeciesName*] = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Scope

Aria Material

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bulk Species [*{of}* *SpeciesName*] = Global (Global_Name = *global_name* | [Component = *component*])

Scope

Aria Material

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Bulk Species [*{of}* *SpeciesName*] = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
vars	"string"1["string"2["string"3["string"4["string"5]]]]	–
i0	"string"	–
i1	"string"	–
i2	"string"	–
x	"string"	–
y	"string"	–
z	"string"	–

Polynomial

Syntax

Bulk Species [*{of}* *SpeciesName*] = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bulk Species [*{of}* *SpeciesName*] = Scalar_String_Function F = *f*

Scope

Aria Material

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be

a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	"string"	–

User_Field

Syntax

Bulk Species [*{of}* *SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bulk Species [*{of}* *SpeciesName*] = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Scope

Aria Material

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Bulk Species [*{of} SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	“string”	–
plugin_parameters	“string”	–

Bulk Viscosity

Constant

Syntax

Bulk Viscosity [*{of} SpeciesName*] = Constant *{kappa | value}* = *kappa*

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
kappa	real	–

Copied

Syntax

Bulk Viscosity [*{of}* *SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	“string”	–

Curing_Foam

Syntax

Bulk Viscosity [*{of}* *SpeciesName*] = Curing_Foam (Mu_A = *mu_a* | T_Mu = *t_mu* | N_Mu = *n_mu* | E_Mu = *e_mu* | R = *r* | Ksi_C = *ksi_c* | M_Ksi_C = *m_ksi_c* | [Extent_Subindex = *extent_subindex*] | [Extent_Species_Name = *extent_species_name*] | [Extent_Species_Phase = *extent_species_phase*])

Scope

Aria Material

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
mu_a	real	–
t_mu	real	–
n_mu	real	–
e_mu	real	–
r	real	–
ksi_c	real	–
m_ksi_c	real	–
extent_subindex	integer	-1
extent_species_name	"string"	–
extent_species_phase	"string"	–

Exponential

Syntax

Bulk Viscosity [*{of}* *SpeciesName*] = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Scope

Aria Material

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bulk Viscosity [*{of}* *SpeciesName*] = Global (Global_Name = *global_name* | [Component = *component*])

Scope

Aria Material

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
global_name	“string”	–
component	integer	0

Interpolated_Phase_Average

Syntax

Bulk Viscosity [*{of}* *SpeciesName*] = Interpolated_Phase_Average ([Subindex_A = *subindex_a*] | [Subindex_B = *subindex_b*])

Scope

Aria Material

Summary

Phase-averaged values

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
subindex_a	integer	-1
subindex_b	integer	-1

Mass_Average

Syntax

Bulk Viscosity [*{of} SpeciesName*] = Mass_Average ()

Scope

Aria Material

Summary

Bulk Viscosity computed from mass average of species viscosities

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–

Nd_Table

Syntax

Bulk Viscosity [*{of} SpeciesName*] = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Phase_Average

Syntax

Bulk Viscosity [*{of}* *SpeciesName*] = Phase_Average ([Subindex_A = *subindex_a*] | [Subindex_B = *subindex_b*])

Scope

Aria Material

Summary

Phase-averaged values

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
subindex_a	integer	-1
subindex_b	integer	-1

Polynomial

Syntax

Bulk Viscosity [*{of}* *SpeciesName*] = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bulk Viscosity [*{of} SpeciesName*] = Scalar_String_Function F = *f*

Scope

Aria Material

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	"string"	–

User_Field

Syntax

Bulk Viscosity [*{of}* *SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bulk Viscosity [*{of}* *SpeciesName*] = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Scope

Aria Material

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Bulk Viscosity [*{of}* *SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	“string”	–
plugin_parameters	“string”	–

Volume_Average

Syntax

Bulk Viscosity [*{of}* *SpeciesName*] = Volume_Average ()

Scope

Aria Material

Summary

Bulk Viscosity computed from volume average of species viscosities, over a material phase

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–

Burn Front Width

Constant

Syntax

Burn Front Width [*{of}* *SpeciesName*] = Constant (Width = *width* | [Use_Crossing_Time = *use_crossing_time*])

Scope

Aria Material

Summary

Specification of a constant width burn front

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
width	real	–
use_crossing_time	integer	–

Burn Heat Release

Constant

Syntax

Burn Heat Release [*{of}* *SpeciesName*] = Constant Value = *value*

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
value	real	–

Copied

Syntax

Burn Heat Release [*{of}* *SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	“string”	–

Exponential

Syntax

Burn Heat Release [*{of}* *SpeciesName*] = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Scope

Aria Material

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	“string”	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Burn Heat Release [*{of}* *SpeciesName*] = Global (Global_Name = *global_name* | [Component = *component*])

Scope

Aria Material

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
global_name	“string”	–
component	integer	0

Nd_Table

Syntax

Burn Heat Release [*{of}* *SpeciesName*] = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Burn Heat Release [*{of}* *SpeciesName*] = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Burn Heat Release [*{of}* *SpeciesName*] = Scalar_String_Function F = *f*

Scope

Aria Material

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be

a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	"string"	–

User_Field

Syntax

Burn Heat Release [*{of}* *SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Burn Heat Release [*{of}* *SpeciesName*] = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Scope

Aria Material

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Burn Heat Release [*{of} SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	“string”	–
plugin_parameters	“string”	–

Burn Ignition

Chemeq

Syntax

Burn Ignition [*{of} SpeciesName*] = Chemeq ()

Scope

Aria Material

Summary

Use the status of the chemeq block to determine when burn fronts become

active. Chemeq is deactivated when either the temperature or rate of temperature criteria are met, which will then ignite a burn front. Burn front ignition controls when burn fronts become active, while initiation line commands refer to where the burn fronts will be initialized.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–

Burn Initiation

Chemeq

Syntax

Burn Initiation [*{of}* *SpeciesName*] = Chemeq ([Offset = *offset*] | [Min_Ignition_Spacing = *min_ignition_spacing*] | [Hard_Start_Ignition = *hard_start_ignition*])

Scope

Aria Material

Summary

Chemeq activated burn front. If min_ignition_spacing is provided then multiple burn initiations are supported, otherwise only one initiation per block is supported. Pick a minimum ignition spacing (d) that is several times larger than your interface width ($d > 3w$), and that provides $Da >> 10$, where $Da = \frac{v_b d}{\alpha}$ (α = thermal diffusivity and v_b = burn speed). By default, the level set is shifted back by $w/2$ on ignition so the burn starts smoothly. To disable this and use a hard start (no shifting of burn front) set hard_start_ignition=1. Chemeq burn initiation also requires Chemeq burn ignition, which is created by default if not specified. The optional argument offset specifies an offset in temperature or temperature rate, such that the level set field is initialized at the chemeq deactivation temperature or rate minus the offset. Ignition is not affected by the offset.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
offset	real	0
min_ignition_spacing	real	0
hard_start_ignition	integer	0

Temperature

Syntax

Burn Initiation [*{of}* *SpeciesName*] = Temperature (Tign = *Tign* | [Min_Ignition_Spacing = *min_ignition_spacing*] | [Hard_Start_Ignition = *hard_start_ignition*])

Scope

Aria Material

Summary

Thermally activated burn front. If min_ignition_spacing is provided then multiple burn initiations are supported, otherwise only one initiation per block is supported. Pick a minimum ignition spacing (d) that is several times larger than your interface width ($d > 3w$), and that provides $Da >> 10$, where $Da = \frac{v_b d}{\alpha}$ (α = thermal diffusivity and v_b = burn speed) By default, the level set is shifted back by $w/2$ on ignition so the burn starts smoothly. To disable this and use a hard start (no shifting of burn front) set hard_start_ignition=1. Burn initiation is used to specify where the burn front occurs, while burn ignition commands specify when to enable burn fronts. If no burn ignition is specified, the ignition method is the same as the initiation

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Tign	real	–
min_ignition_spacing	real	0
hard_start_ignition	integer	0

Burn Speed

Constant

Syntax

Burn Speed [*{of}* *SpeciesName*] = Constant Value = *Value*

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Value	real	–

Exponential

Syntax

Burn Speed [*{of}* *SpeciesName*] = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Scope

Aria Material

Summary

Value from an exponential function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Burn Speed [*{of}* *SpeciesName*] = Global (Global_Name = *global_name* | [Component = *component*])

Scope

Aria Material

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
global_name	"string"	–
component	integer	0

Polynomial

Syntax

Burn Speed [*{of}* *SpeciesName*] = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Scope

Aria Material

Summary

Value from a polynomial function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Burn Speed [*{of}* *SpeciesName*] = Scalar_String_Function F = *f*

Scope

Aria Material

Summary

Value from a scalar string function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	"string"	–

User_Field

Syntax

Burn Speed [*{of}* *SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Burn Speed [*{of}* *SpeciesName*] = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Scope

Aria Material

Summary

User Function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

C

Capillary Pressure

Brooks_Corey

Syntax

Capillary Pressure [*{of}* *SpeciesName*] = Brooks_Corey [Wetting_Phase = *wetting_phase*]

Scope

Aria Material

Summary

Capillary pressure computed based on the Brooks-Corey model

Description

Residual saturations for liquid and gas are S_{lr} and S_{gr}.

Hence the maximum possible liquid saturation is $S_{lm} = 1 - S_{gr}$ and the effective saturation is $S_e = (S_l - S_{lr})/(S_{lm} - S_{lr})$ where S_l is the wetting phase saturation. The pore size distribution index for the media is λ.

Using a cutoff saturation $S_{ec} = 0.01$ and if $S_e < S_{ec}$ let $S_e = S_{ec}$. The capillary pressure is evaluated as $p_c = p_e S_e^{-1/\lambda}$ where p_e is the entry pressure.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
wetting_phase	“string”	WETTING_PHASE

Computed

Syntax

Capillary Pressure [*{of} SpeciesName*] = Computed ([Wetting_Phase = *wetting_phase*] | [Nonwetting_Phase = *nonwetting_phase*])

Scope

Aria Material

Summary

Capillary pressure computed based on the pressure difference between the non-wetting and wetting phases, each of which may be specified as parameters.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
wetting_phase	"string"	WETTING_PHASE
nonwetting_phase	"string"	NONWETTING_PHASE

Constant

Syntax

Capillary Pressure [*{of} SpeciesName*] = Constant Value = *value*

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
value	real	–

Copied

Syntax

Capillary Pressure [*{of} SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	"string"	–

Encore_Saturation

Syntax

Capillary Pressure [*{of}* *SpeciesName*] = Encore_Saturation (Name = *name* | [Result_Name = *result_name*] | [Eval_Type = *eval_type*])

Scope

Aria Material

Summary

A Van Genuchten model for air/water capillary pressure based on an encore function for saturation

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
result_name	"string"	–
eval_type	"string"	–

Exponential

Syntax

Capillary Pressure [*{of}* *SpeciesName*] = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Scope

Aria Material

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Capillary Pressure [*{of} SpeciesName*] = Global (Global_Name = *global_name* | [Component = *component*])

Scope

Aria Material

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Capillary Pressure [*{of} SpeciesName*] = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Capillary Pressure [*{of}* *SpeciesName*] = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Capillary Pressure [*{of} SpeciesName*] = Scalar_String_Function F = *f*

Scope

Aria Material

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	"string"	–

Udell_Cubic

Syntax

Capillary Pressure [*{of}* *SpeciesName*] = Udell_Cubic (Sigma = *sigma* | C1 = *c1* | C2 = *c2* | C3 = *c3* | Residual_Wetting_Phase_Saturation = *residual_wetting_phase_saturation* | Residual_Nonwetting_Phase_Saturation = *residual_nonwetting_phase_saturation* | [Wetting_Phase = *wetting_phase*])

Scope

Aria Material

Summary

A Udell cubic model for air/water capillary pressure

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
sigma	real	–
c1	real	–
c2	real	–
c3	real	–
residual_wetting_phase_saturation	real	–
residual_nonwetting_phase_saturation	real	–
wetting_phase	“string”	WETTING_PHASE

User_Field

Syntax

Capillary Pressure [*{of}* *SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
scaling	real	1
global_var	“string”	–
time_function	“string”	–

User_Function

Syntax

Capillary Pressure [*{of}* *SpeciesName*] = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Scope

Aria Material

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Capillary Pressure [*{of}* *SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	“string”	–
plugin_parameters	“string”	–

Van_Genuchten

Syntax

Capillary Pressure [*{of}* *SpeciesName*] = Van_Genuchten ([Wetting_Phase = *wetting_phase*] | [Beta_Field = *beta_field*] | [Pcref_Field = *pcref_field*] | [Sgr_Field = *sgr_field*] | [Slr_Field = *slr_field*] | [Beta = *beta*] | [Reference_Capillary_Pressure = *reference_capillary_pressure*] | [Residual_Wetting_Phase_Saturation = *residual_wetting_phase_saturation*] | [Residual_Nonwetting_Phase_Saturation = *residual_nonwetting_phase_saturation*])

Scope

Aria Material

Summary

A Van Genuchten model for air/water capillary pressure

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
wetting_phase	“string”	WETTING_PHASE
beta_field	“string”	–
pcref_field	“string”	–
sgr_field	“string”	–
slr_field	“string”	–
beta	real	0
reference_capillary_pressure	real	0
residual_wetting_phase_saturation	real	0
residual_nonwetting_phase_saturation	real	0

Van_Genuchten_Bc

Syntax

Capillary Pressure [*{of} SpeciesName*] = Van_Genuchten_Bc
Wetting_Saturation_Expression_Name = *wetting_saturation_expression_name*

Scope

Aria Material

Summary

A Van Genuchten model for air/water capillary pressure based on an arbitrary function for wetting phase saturation

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
wetting_saturation_expression_name	"string"	–

Characteristic Length

Correlation

Syntax

Characteristic Length [*{of} SpeciesName*] = Correlation (Dh = *Dh* |
[Bulk_Node_Interface = *bulk_node_interface*])

Scope

Aria Material

Summary

Correlation characteristic length with constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Dh	real	–
bulk_node_interface	"string"	–

Correlation_Bar_Perimeter

Syntax

Characteristic Length [*{of}* *SpeciesName*] = Correlation_Bar_Perimeter
[Bulk_Node_Interface = *bulk_node_interface*]

Scope

Aria Material

Summary

Correlation characteristic length with bar perimeter model

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
bulk_node_interface	"string"	–

Correlation_Hydraulic_Diameter

Syntax

Characteristic Length [*{of}* *SpeciesName*] = Correlation_Hydraulic_Diameter
[Bulk_Node_Interface = *bulk_node_interface*]

Scope

Aria Material

Summary

Correlation characteristic length with hydraulic diameter model

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
bulk_node_interface	"string"	–

Correlation_Wetted_Perimeter

Syntax

Characteristic Length [*{of}* *SpeciesName*] = Correlation_Wetted_Perimeter
[Bulk_Node_Interface = *bulk_node_interface*]

Scope

Aria Material

Summary

Correlation characteristic length with wetted perimeter model

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
bulk_node_interface	“string”	–

Charge Density

Constant

Syntax

Charge Density [*{of}* *SpeciesName*] = Constant Value = *value*

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
value	real	–

Copied

Syntax

Charge Density [*{of}* *SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	“string”	–

Exponential

Syntax

Charge Density [*{of}* *SpeciesName*] = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Scope

Aria Material

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Charge Density [*{of}* *SpeciesName*] = Global (Global_Name = *global_name* | [Component = *component*])

Scope

Aria Material

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Charge Density [*{of}* *SpeciesName*] = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Charge Density [*{of}* *SpeciesName*] = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Charge Density [*{of}* *SpeciesName*] = Scalar_String_Function F = *f*

Scope

Aria Material

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	"string"	–

User_Field

Syntax

Charge Density [*{of}* *SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Charge Density [*{of}* *SpeciesName*] = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Scope

Aria Material

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Charge Density [*{of}* *SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	“string”	–
plugin_parameters	“string”	–

Charge Density Diffusive Flux

Constant

Syntax

Charge Density Diffusive Flux [*{of}* *SpeciesName*] = Constant ([Valuex = *valuex*] | [Valuey = *valuey*] | [Valuez = *valuez*])

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
valuex	real	0
valuey	real	0
valuez	real	0

Copied

Syntax

Charge Density Diffusive Flux [*{of}* *SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	“string”	–

Ohms_Law

Syntax

Charge Density Diffusive Flux [*{of}* *SpeciesName*] = Ohms_Law ()

Scope

Aria Material

Summary

Ohm’s law version of charge density diffusive flux

Description

The diffusive flux is proportional to the gradient of the voltage:

$$\mathbf{J} = -\sigma \nabla \mathbf{V},$$

where σ is the electrical conductivity, and V is the voltage.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–

Scalar_Vector_Product

Syntax

Charge Density Diffusive Flux [*{of}* *SpeciesName*] = Scalar_Vector_Product (Vector = *vector* | [Scalars = *scalars*] | [Multiplier = *multiplier*])

Scope

Aria Material

Summary

Scalar vector product

Description

Multiplies a vector by a list of provided scalars (variables) and a multiplier (constant)

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
vector	“string”	–
scalars	“string”	–
multiplier	real	1

Tensor_Vector_Product

Syntax

Charge Density Diffusive Flux [*{of}* *SpeciesName*] = Tensor_Vector_Product (Vector = *vector* | Tensor = *tensor* | [Scale = *scale*])

Scope

Aria Material

Summary

Tensor vector product

Description

Calculates a vector resulting from a tensor-vector product with an optional scaling factor.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
vector	"string"	–
tensor	"string"	–
scale	real	1

User_Field

Syntax

Charge Density Diffusive Flux [*{of}* *SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Vector from User Field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Charge Density Diffusive Flux [*{of}* *SpeciesName*] = User_Function ([Name_X = *name_x*] | [Name_Y = *name_y*] | [Name_Z = *name_z*] | X = *x* | [Multiplier = *multiplier*])

Scope

Aria Material

Summary

Value from a Vector User Function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name_x	"string"	–
name_y	"string"	–
name_z	"string"	–
x	"string"	–
multiplier	real	1

User_Plugin

Syntax

Charge Density Diffusive Flux [*{of}* *SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	"string"	–
plugin_parameters	"string"	–

Vector_String_Function

Syntax

Charge Density Diffusive Flux [*{of}* *SpeciesName*] = Vector_String_Function (F_X = *f_x* | F_Y = *f_y* | [F_Z = *f_z*])

Scope

Aria Material

Summary

Value from a string function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f_x	"string"	–
f_y	"string"	–
f_z	"string"	–

Chemical Potential

From_Equilibrium_Potential

Syntax

Chemical Potential [*{of} SpeciesName*] = From_Equilibrium_Potential [F = *f*]

Scope

Aria Material

Summary

Chemical potential from the equilibrium voltage of an electrochemical reaction. This model assumes that it is a single electron intercalation reaction and that only differences in the chemical potential are relevant, not the absolute magnitude. Therefore $\mu = -F\phi_{eq}$.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	real	9.649e+04

Ideal_Solution

Syntax

Chemical Potential [*{of} SpeciesName*] = Ideal_Solution ([R = *r*] | [Temperature_Material_Phase = *temperature_material_phase*])

Scope

Aria Material

Summary

Ideal solution chemical potential, $\mu = RT\log(C)$.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
r	real	8.314
temperature_material_phase	"string"	–

Isotropic_Mesh_Stress

Syntax

Chemical Potential [*{of}* *SpeciesName*] = Isotropic_Mesh_Stress Strain_Coeff = *strain_coeff*

Scope

Aria Material

Summary

Contribution to the chemical potential from species strain. Assumes that the strain is isotropic so $\mu = \Omega\sigma_{kk}/3$ where Ω is the volumetric expansion coefficient and σ is the stress tensor. This model uses the mesh stress provided by a projection equation in order to recover stress gradients.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
strain_coeff	real	–

Contact Electrical Conductance Coefficient

Calore_User_Sub

Syntax

Contact Electrical Conductance Coefficient [*{of}* *SpeciesName*] = Calore_User_Sub (Name = *name* \| Type = *type* \| [Multiplier = *multiplier*] \| [Material_Data_Block = *material_data_block*] \| [Data = *data*] \| [Scaling_Field = *scaling_field*])

Scope

Aria Material

Summary

Values from a Calore user subroutine

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
type	“string”	–
multiplier	real	1
material_data_block	“string”	–
data	“string”	–
scaling_field	“string”	–

Constant

Syntax

Contact Electrical Conductance Coefficient [*{of}* *SpeciesName*] = Constant *{h | value}* = *h*

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
h	real	–

Copied

Syntax

Contact Electrical Conductance Coefficient [*{of}* *SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	“string”	–

Exponential

Syntax

Contact Electrical Conductance Coefficient [*{of}* *SpeciesName*] = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Scope

Aria Material

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Contact Electrical Conductance Coefficient [*{of}* *SpeciesName*] = Global (Global_Name = *global_name* | [Component = *component*])

Scope

Aria Material

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Contact Electrical Conductance Coefficient [*{of}* *SpeciesName*] = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Contact Electrical Conductance Coefficient [*{of}* *SpeciesName*] = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Contact Electrical Conductance Coefficient [*{of} SpeciesName*] = Scalar_String_Function F = *f*

Scope

Aria Material

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	"string"	–

User_Field

Syntax

Contact Electrical Conductance Coefficient [*{of}* *SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Contact Electrical Conductance Coefficient [*{of}* *SpeciesName*] = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Scope

Aria Material

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1
toggle	"string"	–

User_Plugin

Syntax

Contact Electrical Conductance Coefficient [*{of} SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	"string"	–
plugin_parameters	"string"	–

Contact Gap Temperature

Average

Syntax

Contact Gap Temperature ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) = Average [Equations = *Equations. . .*]

Scope

Aria Material

Summary

Average temperature across the interface

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Equations	string. . .	–

Constant

Syntax

Contact Gap Temperature ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) = Constant (*{t | value}* = *t* | [Equations = *Equations. . .*])

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
t	real	–
Equations	string. . .	–

Copied

Syntax

Contact Gap Temperature ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) = Copied (Source = *source* | [Equations = *Equations. . .*])

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
source	"string"	–
Equations	string. . .	–

Exponential

Syntax

Contact Gap Temperature ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent* | [Equations = *Equations. . .*])

Scope

Aria Material

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–
Equations	string. . .	–

Global

Syntax

Contact Gap Temperature ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) = Global (Global_Name = *global_name* | [Component = *component*] | [Equations = *Equations. . .*])

Scope

Aria Material

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
global_name	"string"	–
component	integer	0
Equations	string. . .	–

Nd_Table

Syntax

Contact Gap Temperature ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*] | [Equations = *Equations. . .*])

Scope

Aria Material

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–
Equations	string. . .	–

Polynomial

Syntax

Contact Gap Temperature ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*] | [Equations = *Equations. . .*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0
Equations	string. . .	–

Scalar_String_Function

Syntax

Contact Gap Temperature ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) = Scalar_String_Function (F = *f* | [Equations = *Equations. . .*])

Scope

Aria Material

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
f	"string"	–
Equations	string. . .	–

User_Field

Syntax

Contact Gap Temperature ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*] | [Equations = *Equations. . .*])

Scope

Aria Material

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–
Equations	string. . .	–

User_Function

Syntax

Contact Gap Temperature ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*] | [Equations = *Equations. . .*])

Scope

Aria Material

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1
toggle	"string"	–
Equations	string. . .	–

User_Plugin

Syntax

Contact Gap Temperature ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) = User_Plugin (Name = *Name* | [*plugin_parameters*] | [Equations = *Equations. . .*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Name	"string"	–
plugin_parameters	"string"	–
Equations	string. . .	–

Contact Gap Voltage

Average

Syntax

Contact Gap Voltage ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) = Average [Equations = *Equations. . .*]

Scope

Aria Material

Summary

Average voltage across the interface

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Equations	string. . .	–

Constant

Syntax

Contact Gap Voltage ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) = Constant (*{v | value}* = *v* | [Equations = *Equations. . .*])

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
v	real	–
Equations	string. . .	–

Copied

Syntax

Contact Gap Voltage ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) = Copied (Source = *source* | [Equations = *Equations. . .*])

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
source	“string”	–
Equations	string. . .	–

Exponential

Syntax

Contact Gap Voltage ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent* | [Equations = *Equations. . .*])

Scope

Aria Material

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–
Equations	string. . .	–

Global

Syntax

Contact Gap Voltage ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) = Global (Global_Name = *global_name* | [Component = *component*] | [Equations = *Equations. . .*])

Scope

Aria Material

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
global_name	"string"	–
component	integer	0
Equations	string. . .	–

Nd_Table

Syntax

Contact Gap Voltage ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*] | [Equations = *Equations. . .*])

Scope

Aria Material

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–
Equations	string. . .	–

Polynomial

Syntax

Contact Gap Voltage ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*] | [Equations = *Equations. . .*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0
Equations	string...	–

Scalar_String_Function

Syntax

Contact Gap Voltage ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) = Scalar_String_Function (F = *f* | [Equations = *Equations...*])

Scope

Aria Material

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
f	"string"	–
Equations	string. . .	–

User_Field

Syntax

Contact Gap Voltage ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*] | [Equations = *Equations. . .*])

Scope

Aria Material

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–
Equations	string. . .	–

User_Function

Syntax

Contact Gap Voltage ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*] | [Equations = *Equations. . .*])

Scope

Aria Material

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1
toggle	"string"	–
Equations	string. . .	–

User_Plugin

Syntax

Contact Gap Voltage ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) = User_Plugin (Name = *Name* | [*plugin_parameters*] | [Equations = *Equations. . .*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Name	“string”	–
plugin_parameters	“string”	–
Equations	string. . .	–

Contact Heat Transfer Coefficient

Calore_User_Sub

Syntax

Contact Heat Transfer Coefficient [*{of}* *SpeciesName*] = Calore_User_Sub (Name = *name* | Type = *type* | [Multiplier = *multiplier*] | [Material_Data_Block = *material_data_block*] | [Data = *data*] | [Scaling_Field = *scaling_field*])

Scope

Aria Material

Summary

Values from a Calore user subroutine

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
type	“string”	–
multiplier	real	1
material_data_block	“string”	–
data	“string”	–
scaling_field	“string”	–

Constant

Syntax

Contact Heat Transfer Coefficient [*{of}* *SpeciesName*] = Constant *{h | value}* = *h*

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
h	real	–

Copied

Syntax

Contact Heat Transfer Coefficient [*{of}* *SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	“string”	–

Exponential

Syntax

Contact Heat Transfer Coefficient [*{of}* *SpeciesName*] = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Scope

Aria Material

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Contact Heat Transfer Coefficient [*{of} SpeciesName*] = Global (Global_Name = *global_name* | [Component = *component*])

Scope

Aria Material

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Contact Heat Transfer Coefficient [*{of} SpeciesName*] = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Contact Heat Transfer Coefficient [*{of} SpeciesName*] = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Contact Heat Transfer Coefficient [*{of}* *SpeciesName*] = Scalar_String_Function
F = *f*

Scope

Aria Material

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	"string"	–

User_Field

Syntax

Contact Heat Transfer Coefficient [*{of} SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Contact Heat Transfer Coefficient [*{of} SpeciesName*] = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Scope

Aria Material

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1
toggle	"string"	–

User_Plugin

Syntax

Contact Heat Transfer Coefficient [*{of} SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	"string"	–
plugin_parameters	"string"	–

Contact Surface Pressure

Average

Syntax

Contact Surface Pressure ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) = Average (Source = *source* | [Equations = *Equations. . .*])

Scope

Aria Material

Summary

Average pressure across the interface

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
source	"string"	–
Equations	string. . .	–

Constant

Syntax

Contact Surface Pressure ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) = Constant (*{p | value}* = *p* | [Equations = *Equations. . .*])

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
p	real	–
Equations	string. . .	–

Copied

Syntax

Contact Surface Pressure ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) = Copied (Source = *source* | [Equations = *Equations. . .*])

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
source	"string"	–
Equations	string. . .	–

Either_Or

Syntax

Contact Surface Pressure ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) = Either_Or (Source = *source* | [Equations = *Equations. . .*])

Scope

Aria Material

Summary

Select expression found on either side of contact interface.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
source	"string"	–
Equations	string. . .	–

Exponential

Syntax

Contact Surface Pressure ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent* | [Equations = *Equations. . .*])

Scope

Aria Material

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–
Equations	string. . .	–

Global

Syntax

Contact Surface Pressure ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) = Global (Global_Name = *global_name* | [Component = *component*] | [Equations = *Equations. . .*])

Scope

Aria Material

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
global_name	"string"	–
component	integer	0
Equations	string. . .	–

Nd_Table

Syntax

Contact Surface Pressure ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*] | [Equations = *Equations. . .*])

Scope

Aria Material

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–
Equations	string. . .	–

Polynomial

Syntax

Contact Surface Pressure ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*] | [Equations = *Equations. . .*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0
Equations	string. . .	–

Scalar_String_Function

Syntax

Contact Surface Pressure ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) = Scalar_String_Function (F = *f* | [Equations = *Equations. . .*])

Scope

Aria Material

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
f	"string"	–
Equations	string. . .	–

User_Field

Syntax

Contact Surface Pressure ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*] | [Equations = *Equations. . .*])

Scope

Aria Material

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–
Equations	string. . .	–

User_Function

Syntax

Contact Surface Pressure ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*] | [Equations = *Equations. . .*])

Scope

Aria Material

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1
toggle	"string"	–
Equations	string. . .	–

User_Plugin

Syntax

Contact Surface Pressure ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) = User_Plugin (Name = *Name* | [*plugin_parameters*] | [Equations = *Equations*. . .])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Name	"string"	–
plugin_parameters	"string"	–
Equations	string. . .	–

Continuity Diffusion Coefficient

Constant

Syntax

Continuity Diffusion Coefficient [*{of}* *SpeciesName*] = Constant Value = *value*

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
value	real	–

Copied

Syntax

Continuity Diffusion Coefficient [*{of}* *SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	"string"	–

Exponential

Syntax

Continuity Diffusion Coefficient [*{of} SpeciesName*] = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Scope

Aria Material

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Continuity Diffusion Coefficient [*{of} SpeciesName*] = Global (Global_Name = *global_name* | [Component = *component*])

Scope

Aria Material

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Continuity Diffusion Coefficient [*{of}* *SpeciesName*] = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Continuity Diffusion Coefficient [*{of}* *SpeciesName*] = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Continuity Diffusion Coefficient [*{of}* *SpeciesName*] = Scalar_String_Function F = *f*

Scope

Aria Material

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	"string"	–

Shakib

Syntax

Continuity Diffusion Coefficient [*{of} SpeciesName*] = Shakib ([Multiplier = *multiplier*] | [Use_Diffusion = *use_diffusion*] | [Use_Advection = *use_advection*] | [Use_Time = *use_time*])

Scope

Aria Material

Summary

Pressure POISSON with diffusion coefficient given by Shakib tau.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
multiplier	real	1
use_diffusion	integer	–
use_advection	integer	–
use_time	integer	–

User_Field

Syntax

Continuity Diffusion Coefficient [*{of} SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
scaling	real	1
global_var	“string”	–
time_function	“string”	–

User_Function

Syntax

Continuity Diffusion Coefficient [*{of}* *SpeciesName*] = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Scope

Aria Material

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1
toggle	"string"	–

User_Plugin

Syntax

Continuity Diffusion Coefficient [*{of}* *SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	“string”	–
plugin_parameters	“string”	–

Continuity Diffusive Flux

Basic

Syntax

Continuity Diffusive Flux [*{of}* *SpeciesName*] = Basic ()

Scope

Aria Material

Summary

Basic model of continuity diffusive flux

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–

Pressure_Poisson

Syntax

Continuity Diffusive Flux [*{of}* *SpeciesName*] = Pressure_Poisson ()

Scope

Aria Material

Summary

This model has been deprecated and removed from the code.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–

Continuity Face Stabilization Scaling

Default

Syntax

Continuity Face Stabilization Scaling ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) = Default ()

Scope

Aria Material

Summary

Default face stabilization scaling for continuity equation

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–

Cte

Constant

Syntax

Cte [*{of} SpeciesName*] = Constant *{cte | value}* = *cte*

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
cte	real	–

Copied

Syntax

Cte [*{of}* *SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	"string"	–

Exponential

Syntax

Cte [*{of}* *SpeciesName*] = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Scope

Aria Material

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Cte [*{of} SpeciesName*] = Global (Global_Name = *global_name* | [Component = *component*])

Scope

Aria Material

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
global_name	“string”	–
component	integer	0

Nd_Table

Syntax

Cte [*{of} SpeciesName*] = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Cte [*{of}* *SpeciesName*] = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Cte [*{of}* *SpeciesName*] = Scalar_String_Function F = *f*

Scope

Aria Material

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be

a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	"string"	–

User_Field

Syntax

Cte [*{of}* *SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Cte [*{of}* *SpeciesName*] = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Scope

Aria Material

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1
toggle	"string"	–

User_Plugin

Syntax

Cte [*{of}* *SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	"string"	–
plugin_parameters	"string"	–

Current Density

Basic

Syntax

Current Density [*{of}* *SpeciesName*] = Basic ()

Scope

Aria Material

Summary

Basic current density

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–

Constant

Syntax

Current Density [*{of}* *SpeciesName*] = Constant ([Valuex = *valuex*] | [Valuey = *valuey*] | [Valuez = *valuez*])

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
valuex	real	0
valuey	real	0
valuez	real	0

Copied

Syntax

Current Density [*{of}* *SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	“string”	–

Curl_B

Syntax

Current Density [*{of} SpeciesName*] = Curl_B ()

Scope

Aria Material

Summary

Calculate current density from magnetic field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–

Dielectric_Displacement

Syntax

Current Density [*{of} SpeciesName*] = Dielectric_Displacement ()

Scope

Aria Material

Summary

Dielectric displacement current density

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–

Electrolyte

Syntax

Current Density [*{of} SpeciesName*] = Electrolyte [F = *f*]

Scope

Aria Material

Summary

Electrolyte current density

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	real	9.649e+04

Mhd

Syntax

Current Density [*{of}* *SpeciesName*] = Mhd ([Magnetic_Field = *magnetic_field*] | [No_Adv = *no_adv*])

Scope

Aria Material

Summary

Magnetohydrodynamic current density

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
magnetic_field	“string”	–
no_adv	“string”	–

Ohms_Law

Syntax

Current Density [*{of}* *SpeciesName*] = Ohms_Law ()

Scope

Aria Material

Summary

Ohms law current density

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–

Scalar_Vector_Product

Syntax

Current Density [*{of}* *SpeciesName*] = Scalar_Vector_Product (Vector = *vector* | [Scalars = *scalars*] | [Multiplier = *multiplier*])

Scope

Aria Material

Summary

Scalar vector product

Description

Multiplies a vector by a list of provided scalars (variables) and a multiplier (constant)

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
vector	"string"	–
scalars	"string"	–
multiplier	real	1

Tensor_Vector_Product

Syntax

Current Density [*{of}* *SpeciesName*] = Tensor_Vector_Product (Vector = *vector* | Tensor = *tensor* | [Scale = *scale*])

Scope

Aria Material

Summary

Tensor vector product

Description

Calculates a vector resulting from a tensor-vector product with an optional scaling factor.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
vector	"string"	–
tensor	"string"	–
scale	real	1

Thermoelectric

Syntax

Current Density [*{of}* *SpeciesName*] = Thermoelectric ()

Scope

Aria Material

Summary

Thermoelectric current density

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–

User_Field

Syntax

Current Density [*{of}* *SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Vector from User Field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Current Density [*{of}* *SpeciesName*] = User_Function ([Name_X = *name_x*] | [Name_Y = *name_y*] | [Name_Z = *name_z*] | X = *x* | [Multiplier = *multiplier*])

Scope

Aria Material

Summary

Value from a Vector User Function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name_x	"string"	–
name_y	"string"	–
name_z	"string"	–
x	"string"	–
multiplier	real	1

User_Plugin

Syntax

Current Density [*{of}* *SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	"string"	–
plugin_parameters	"string"	–

Vector_String_Function

Syntax

Current Density [*{of}* *SpeciesName*] = Vector_String_Function (F_X = *f_x* | F_Y = *f_y* | [F_Z = *f_z*])

Scope

Aria Material

Summary

Value from a string function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f_x	“string”	–
f_y	“string”	–
f_z	“string”	–

Current Exchange Density

Constant

Syntax

Current Exchange Density [*{of}* *SpeciesName*] = Constant *{i_0 | value}* = *i_0*

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
i_0	real	–

Copied

Syntax

Current Exchange Density [*{of}* *SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	“string”	–

Exponential

Syntax

Current Exchange Density [*{of}* *SpeciesName*] = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Scope

Aria Material

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Current Exchange Density [*{of}* *SpeciesName*] = Global (Global_Name = *global_name* | [Component = *component*])

Scope

Aria Material

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Current Exchange Density [*{of}* *SpeciesName*] = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Current Exchange Density [*{of}* *SpeciesName*] = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Current Exchange Density [*{of}* *SpeciesName*] = Scalar_String_Function F = *f*

Scope

Aria Material

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	"string"	–

User_Field

Syntax

Current Exchange Density [*{of}* *SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Current Exchange Density [*{of}* *SpeciesName*] = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Scope

Aria Material

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Current Exchange Density [*{of} SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	“string”	–
plugin_parameters	“string”	–

D

Decay Constant

Hdiff

Syntax

Decay Constant [*{of} SpeciesName*] = Hdiff Decay = *decay*

Scope

Aria Material

Summary

Decay constant used for hydrogen transport

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
decay	real	–

Density

Adaptive_Table_Lookup

Syntax

Density [*{of}* *SpeciesName*] = Adaptive_Table_Lookup File_Name = *file_name*

Scope

Aria Material

Summary

An adaptive table lookup of density

Description

Performs bilinear interpolation based on the specified table file using $T + 273.15$ and P as the coordinates.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
file_name	"string"	–

Calore_User_Sub

Syntax

Density [*{of}* *SpeciesName*] = Calore_User_Sub (Name = *name* | Type = *type* | [Multiplier = *multiplier*] | [Material_Data_Block = *material_data_block*] | [Data = *data*] | [Scaling_Field = *scaling_field*])

Scope

Aria Material

Summary

Values from a Calore user subroutine

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
type	"string"	–
multiplier	real	1
material_data_block	"string"	–
data	"string"	–
scaling_field	"string"	–

Cantera

Syntax

Density [*{of}* *SpeciesName*] = Cantera [Poffset = *poffset*]

Scope

Aria Material

Summary

Evaluates the density using the Cantera library.

Description

A valid Cantera XML File must be provided elsewhere in the material definition.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
poffset	real	0

Cantera_Molten_Salt

Syntax

Density [*{of}* *SpeciesName*] = Cantera_Molten_Salt (Densityconversion = *densityConversion* \| Canteraxmlfile = *canteraXMLFile*)

Scope

Aria Material

Summary

This model has been deprecated and removed from the code.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
densityConversion	real	–
canteraXMLFile	"string"	–

Clsm

Syntax

Density [*{of}* *SpeciesName*] = Clsm (Primaryproperty = *primaryProperty* | Secondaryproperty = *secondaryProperty*)

Scope

Aria Material

Summary

Level-Set averaged density.

Description

$$\rho = \phi primaryProperty + (1 - \phi) secondaryProperty$$

where ϕ is the local value of the level set.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
primaryProperty	real	–
secondaryProperty	real	–

Compressible_Boussinesq

Syntax

Density [*{of}* *SpeciesName*] = Compressible_Boussinesq (Ref_Density = *ref_density* | Alpha = *alpha* | Ref_Pressure = *ref_pressure* | [Ref_Temperature = *ref_temperature*])

Scope

Aria Material

Summary

A compressible Boussinesq model for the density

Description

$$\rho = \rho_{\text{ref}} * (1 + \alpha(P - P_{\text{ref}})) + \rho_T$$

If the model does not specify the temperature then $\rho_T = 0$, if it does then the "REF_TEMPERATURE" parameter must be provided for this model, and a model for the isobaric compressibility, β, must also be present in the material definition and $\rho_T = \rho_{\text{ref}} * \beta * (T - T_{\text{ref}})$.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
ref_density	real	–
alpha	real	–
ref_pressure	real	–
ref_temperature	real	0

Concentration_Average

Syntax

Density [*{of}* *SpeciesName*] = Concentration_Average (Fluid_Density = *fluid_density* | Lambda = *lambda*)

Scope

Aria Material

Summary

This model is now deprecated

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
fluid_density	real	–
lambda	real	–

Constant

Syntax

Density [*{of}* *SpeciesName*] = Constant *{rho | value}* = *rho*

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
rho	real	–

Copied

Syntax

Density [*{of}* *SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	“string”	–

Curing_Foam

Syntax

Density [*{of}* *SpeciesName*] = Curing_Foam (Vfrac_Subindex = *vfrac_subindex* | Rho_E = *rho_e* | Rho_F = *rho_f* | R = *r* | Phi_Zero = *phi_zero*)

Scope

Aria Material

Summary

Density model for curing polymer foams.

Description

$$\rho = (\phi_0 - \phi) * \frac{PMW}{RT} + (1 - \phi_0)\rho_e + \phi\rho_f$$

where ϕ is the species concentration corresponding to the “vfrac_subindex” parameter, and ϕ_0, ρ_e, ρ_f, and R are the provided parameters.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
vfrac_subindex	integer	–
rho_e	real	–
rho_f	real	–
r	real	–
phi_zero	real	–

Exp_Decay

Syntax

Density [*{of}* *SpeciesName*] = Exp_Decay (Rho_Initial = *rho_initial* | Rho_Final = *rho_final* | K = *k*)

Scope

Aria Material

Summary

Density as an exponential decay function in time

Description

$\rho = \rho_{\text{final}} + (\rho_{\text{initial}} - \rho_{\text{final}})e^{-kt}$

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
rho_initial	real	–
rho_final	real	–
k	real	–

Exponential

Syntax

Density [*{of}* *SpeciesName*] = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Scope

Aria Material

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Foam_Time_Temp

Syntax

Density [*{of}* *SpeciesName*] = Foam_Time_Temp (Rho_Initial = *rho_initial* | Rho_Final = *rho_final* | C_Param = *c_param* | D_Param = *d_param* | T_Max = *t_max* | [Time_Naught = *time_naught*])

Scope

Aria Material

Summary

Empirical parameterization of density for time and temperature dependent density of an expanding foam.

Description

$$\rho = \left(\rho_i^o - \rho_f^o\right) \exp\left[\frac{-(t - t_i)}{\frac{C}{T^m} - D}\right] + \rho_f^o$$

where $T^m = min(T, T_{max})$.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
rho_initial	real	–
rho_final	real	–
c_param	real	–
d_param	real	–
t_max	real	–
time_naught	real	0

Fortran

Syntax

Density [*{of}* *SpeciesName*] = Fortran ([Multiplier = *multiplier*] | Sub_Name = *sub_name* | [Real_Data = *real_data*] | [Int_Data = *int_data*] | [Resource_Name = *resource_name*] | [Data = *data*])

Scope

Aria Material

Summary

Fortran subroutine density.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
multiplier	real	1
sub_name	"string"	–
real_data	"string"	–
int_data	"string"	–
resource_name	"string"	–
data	"string"	–

From_Bubble_State

Syntax

Density [*{of}* *SpeciesName*] = From_Bubble_State Gas_Species_Name = *gas_species_name*

Scope

Aria Material

Summary

Density from foam bubble model

Description

$$\rho = \frac{MW_g * c_g}{\phi}$$

where MW_g is the molecular weight of the specified gas species, c_g is its concentration, and ϕ is the volume fraction of gas.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
gas_species_name	"string"	–

From_Chemeq

Syntax

Density [*{of}* *SpeciesName*] = From_Chemeq ()

Scope

Aria Material

Summary

Density of one species extracted from ChemEq

Description

$$\rho_i = \text{convert}(c_{i,\text{ChemEq}})$$

where the convert() function evaluates the appropriate conversion from the ChemEq model concentration units to mass density.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–

From_Chemeq_Solids

Syntax

Density [*{of}* *SpeciesName*] = From_Chemeq_Solids [Rhob0 = *rhob0*]

Scope

Aria Material

Summary

Condensed phase density from ChemEq species concentrations for solid species

Description

$$\rho = \Sigma\text{convert}(c_{\text{ChemEq}})$$

where the convert() function evaluates the appropriate conversion from the ChemEq model concentration units to mass density, and the summation is over the solid phase species in the ChemEq model.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
rhob0	real	0

From_Mass_Fraction

Syntax

Density [*{of}* *SpeciesName*] = From_Mass_Fraction ()

Scope

Aria Material

Summary

Species density computed from mass fraction

Description

$$\rho_i = \rho * Y_i$$

where ρ is the overall density, ρ_i the species density, and Y_i the mass fraction of the species. This model should only be used for the density of a species.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–

From_Species

Syntax

Density [*{of}* *SpeciesName*] = From_Species ()

Scope

Aria Material

Summary

Computes an overall density from species concentrations

Description

Computes an overall density by summing the species concentration (c_i) and molecular weight (MW_i) product

$$\rho = \sum_i MW_i c_i$$

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–

From_Volume_Fraction_Gas

Syntax

Density [*{of}* *SpeciesName*] = From_Volume_Fraction_Gas ()

Scope

Aria Material

Summary

Foam density calculated from a weighting with the volume fraction of gas.

Description

$$\rho = \rho_l + \phi_g * (\rho_g - \rho_l)$$

where ϕ_g is the volume fraction gas, ρ_g the gas phase density, and ρ_l the liquid phase density.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–

General_Ideal_Gas

Syntax

Density [*{of}* *SpeciesName*] = General_Ideal_Gas (R = *r* | [Pref = *pref*] | [Poffset = *poffset*])

Scope

Aria Material

Summary

General ideal gas density

Description

$$\rho = \frac{PMW_{\text{mix}}}{RT}$$

where P is either the P_{ref} parameter if provided, or the pressure added to the P_{offset} parameter. MW_{mix} is the mixture molecular weight, R the provided gas constant, and T the temperature.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
r	real	–
pref	real	0
poffset	real	0

General_Ideal_Gas_Extract_Average_Pressure

Syntax

Density [*{of}* *SpeciesName*] = General_Ideal_Gas_Extract_Average_Pressure (R = *r* | [Pref = *pref*] | [Poffset = *poffset*])

Scope

Aria Material

Summary

General ideal gas density that subtracts off the average pressure value as a constant every time step so that the pressure dof stays O(1) throughout the simulation.

Description

See *General_Ideal_Gas* for the equation.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
r	real	–
pref	real	0
poffset	real	0

General_Ideal_Gas_Thermodynamic_Pressure

Syntax

Density [*{of}* *SpeciesName*] = General_Ideal_Gas_Thermodynamic_Pressure (R = *r* | Initial_Thermodynamic_Pressure = *initial_thermodynamic_pressure*)

Scope

Aria Material

Summary

General ideal gas density that uses the thermodynamic pressure in the pressure-density relation, as in the low Mach equations.

Description

See *General_Ideal_Gas* for the equation.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
r	real	–
initial_thermodynamic_pressure	real	–

Global

Syntax

Density [*{of}* *SpeciesName*] = Global (Global_Name = *global_name* | [Component = *component*])

Scope

Aria Material

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
global_name	“string”	–
component	integer	0

Ideal_Gas

Syntax

Density [*{of}* *SpeciesName*] = Ideal_Gas ([P_Ref = *p_ref*] | [T_Ref = *t_ref*] | R = *r*)

Scope

Aria Material

Summary

Ideal gas density

Description

$$\rho = \frac{(P_{\text{ref}} + P)MW_{\text{mix}}}{R(T_{\text{ref}} + T)}$$

where P_{ref} and T_{ref} are reference pressure and temperatures, respectively. T is temperature, P is pressure, MW_{mix} is the mixture molecular weight, R the provided gas constant. In this model $MW_{\text{mix}} = \sum_{i=0}^{n} Y_i MW_{\text{i}}$ where Y_i and MW_{i} are the species mass fraction and molecular weights, respectively for n species.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
p_ref	real	0
t_ref	real	0
r	real	–

Incompressible_Ideal_Gas

Syntax

Density [*{of}* *SpeciesName*] = Incompressible_Ideal_Gas (P_Ref = *p_ref* | [T_Ref = *t_ref*] | R = *r*)

Scope

Aria Material

Summary

Incompressible ideal gas density

Description

$$\rho = \frac{P_{\text{ref}} MW_{\text{mix}}}{R(T + T_{\text{ref}})}$$

MW_{mix} is the mixture molecular weight, R the provided gas constant, and T the temperature.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
p_ref	real	–
t_ref	real	0
r	real	–

Interpolated_Phase_Average

Syntax

Density [*{of}* *SpeciesName*] = Interpolated_Phase_Average ([Subindex_A = *subindex_a*] | [Subindex_B = *subindex_b*])

Scope

Aria Material

Summary

Interpolated level set phase averaged values.

Description

See *Phase_Average*, however in this model the evaluation is done based on nodal values and interpolated to the integration points.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
subindex_a	integer	-1
subindex_b	integer	-1

Mass_Average

Syntax

Density [*{of}* *SpeciesName*] = Mass_Average ()

Scope

Aria Material

Summary

Density as a mass average of the species bulk densities

Description

$$\rho = \sum_i \left(\frac{Y_i}{\rho_{b,i}} \right)^{-1}$$

where Y_i is the mass fraction of each species, and $\rho_{b,i}$ is the bulk density of each species.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–

Mass_Preserving

Syntax

Density [*{of}* *SpeciesName*] = Mass_Preserving Rho = *rho*

Scope

Aria Material

Summary

Density applied as a ratio of original to current element volume.

Description

$$\rho = \frac{\rho_0 V_{e,0}}{V_e}$$

where V_e is the element volume.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
rho	real	–

Mixture_Fraction

Syntax

Density [*{of}* *SpeciesName*] = Mixture_Fraction (Primaryproperty = *primaryProperty* \| Secondaryproperty = *secondaryProperty*)

Scope

Aria Material

Summary

Mixture fraction model for density.

Description

$$\rho = \frac{1}{\frac{Z}{\rho_{\text{primary}}} + \frac{1-Z}{\rho_{\text{secondary}}}}$$

where Z is the mixture fraction, ρ_{primary} the provided primaryProperty, and $\rho_{\text{secondary}}$ the provided secondaryProperty.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
primaryProperty	real	–
secondaryProperty	real	–

Nd_Table

Syntax

Density [*{of}* *SpeciesName*] = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Phase_Average

Syntax

Density [*{of}* *SpeciesName*] = Phase_Average ([Subindex_A = *subindex_a*] | [Subindex_B = *subindex_b*])

Scope

Aria Material

Summary

Level set phase averaged density.

Description

$$\rho = (1 - H(\phi))\rho_A + H(\phi)\rho_B$$

where $H(\phi)$ is the heaviside function evaluated at the level-set variable ϕ, ρ_A is the density in phase A, and ρ_B is the density in phase B.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
subindex_a	integer	-1
subindex_b	integer	-1

Polynomial

Syntax

Density [*{of}* *SpeciesName*] = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	“string”	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Porous_Phase_Average

Syntax

Density [*{of}* *SpeciesName*] = Porous_Phase_Average ([Liquid_Phase_Name = *Liquid_Phase_Name*] | [Gas_Phase_Name = *Gas_Phase_Name*])

Scope

Aria Material

Summary

Density for a multiphase material of two or three phases

Description

For a two-phase (solid and liquid) porous media problem:

$$\rho = (1 - \phi)\rho_s + \phi\rho_l$$

For a three-phase (solid, liquid, gas) porous media problem:

$$\rho = (1 - \phi)\rho_s + \phi S_l \rho_l + \phi(1 - S_l)\rho_g$$

where ϕ is the porosity, ρ_i the density in phase i, and S_l the liquid saturation.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Liquid_Phase_Name	"string"	–
Gas_Phase_Name	"string"	–

Scalar_String_Function

Syntax

Density [*{of}* *SpeciesName*] = Scalar_String_Function F = *f*

Scope

Aria Material

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	"string"	–

Single_Component_Adiabatic_Ideal_Gas

Syntax

Density [*{of}* *SpeciesName*] = Single_Component_Adiabatic_Ideal_Gas ([P_Ref = *p_ref*] | P_0 = *p_0* | Density_0 = *density_0* | Gamma = *gamma*)

Scope

Aria Material

Summary

Adiabatic ideal gas density model for a single species.

Description

$$\rho = \rho_0 * \frac{P}{P_0}^{\frac{1}{\gamma}}$$

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
p_ref	real	0
p_0	real	–
density_0	real	–
gamma	real	–

Single_Component_Ideal_Gas

Syntax

Density [*{of}* *SpeciesName*] = Single_Component_Ideal_Gas ([P_Ref = *p_ref*] | [T_Ref = *t_ref*] | [M = *m*] | R = *r*)

Scope

Aria Material

Summary

Ideal gas density model for a single species.

Description

$$\rho = \frac{P + P_{\text{ref}} M}{R(T + T_{\text{ref}})}$$

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
p_ref	real	0
t_ref	real	0
m	real	1
r	real	–

Stanford

Syntax

Density [*{of}* *SpeciesName*] = Stanford N = *n*

Scope

Aria Material

Summary

Stanford density model.

Description

$$\rho = 2Z(1 - Z)^n$$

where Z is the mixture fraction.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
n	real	–

Sum_All_Bulk_Mass_Densities

Syntax

Density [*{of}* *SpeciesName*] = Sum_All_Bulk_Mass_Densities ()

Scope

Aria Material

Summary

Computes an overall density by summing the bulk mass densities of the gas and solid phases in the material.

Description

$$\rho = \rho_{bulk,gas} + \rho_{bulk,solid}$$

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–

Sum_All_Species

Syntax

Density [*{of}* *SpeciesName*] = Sum_All_Species ()

Scope

Aria Material

Summary

Computes an overall density by summing the densities of all species in the material phase.

Description

$$\rho = \sum_i \rho_i$$

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–

Thermal

Syntax

Density [*{of}* *SpeciesName*] = Thermal ([A = *a*] | [B = *b*] | [C = *c*] | [D = *d*])

Scope

Aria Material

Summary

Density as a cubic polynomial in temperature.

Description

$$\rho = A + bT + cT^2 + dT^3$$

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
a	real	0
b	real	0
c	real	0
d	real	0

User_Field

Syntax

Density [*{of}* *SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Density [*{of}* *SpeciesName*] = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Scope

Aria Material

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1
toggle	"string"	–

User_Plugin

Syntax

Density [*{of} SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	"string"	–
plugin_parameters	"string"	–

Deposit

Binary_State

Syntax

Deposit [*{of} SpeciesName*] = Binary_State Criteria = *criteria*

Scope

Aria Material

Summary

Binary State deposition

Description

Provides an expression that represents the deposited/undeposited state with criteria for deposition. Parameter CRITERIA is a string function criteria to mark an element as deposited.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
criteria	"string"	–

Dispersed Phase Density

Constant

Syntax

Dispersed Phase Density [*{of}* *SpeciesName*] = Constant Value = *value*

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
value	real	–

Copied

Syntax

Dispersed Phase Density [*{of}* *SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	"string"	–

Exponential

Syntax

Dispersed Phase Density [*{of}* *SpeciesName*] = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Scope

Aria Material

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Dispersed Phase Density [*{of}* *SpeciesName*] = Global (Global_Name = *global_name* | [Component = *component*])

Scope

Aria Material

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
global_name	“string”	–
component	integer	0

Nd_Table

Syntax

Dispersed Phase Density [*{of} SpeciesName*] = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Dispersed Phase Density [*{of} SpeciesName*] = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Dispersed Phase Density [*{of} SpeciesName*] = Scalar_String_Function F = *f*

Scope

Aria Material

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	"string"	–

User_Field

Syntax

Dispersed Phase Density [*{of}* *SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Dispersed Phase Density [*{of}* *SpeciesName*] = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Scope

Aria Material

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Dispersed Phase Density [*{of} SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	“string”	–
plugin_parameters	“string”	–

Dispersed Phase Momentum Stress

Newtonian

Syntax

Dispersed Phase Momentum Stress [*{of} SpeciesName*] = Newtonian *{cts | value}* = *cts*

Scope

Aria Material

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
cts	real	–

Dispersed Phase Velocity

Constant

Syntax

Dispersed Phase Velocity [*{of}* *SpeciesName*] = Constant ([Velocity_X = *Velocity_X*] | [Velocity_Y = *Velocity_Y*] | [Velocity_Z = *Velocity_Z*])

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Velocity_X	real	0
Velocity_Y	real	0
Velocity_Z	real	0

Copied

Syntax

Dispersed Phase Velocity [*{of}* *SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	“string”	–

Scalar_Vector_Product

Syntax

Dispersed Phase Velocity [*{of}* *SpeciesName*] = Scalar_Vector_Product (Vector = *vector* | [Scalars = *scalars*] | [Multiplier = *multiplier*])

Scope

Aria Material

Summary

Scalar vector product

Description

Multiplies a vector by a list of provided scalars (variables) and a multiplier (constant)

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
vector	"string"	–
scalars	"string"	–
multiplier	real	1

Tensor_Vector_Product

Syntax

Dispersed Phase Velocity [*{of}* *SpeciesName*] = Tensor_Vector_Product (Vector = *vector* | Tensor = *tensor* | [Scale = *scale*])

Scope

Aria Material

Summary

Tensor vector product

Description

Calculates a vector resulting from a tensor-vector product with an optional scaling factor.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
vector	"string"	–
tensor	"string"	–
scale	real	1

User_Field

Syntax

Dispersed Phase Velocity [*{of}* *SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Vector from User Field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Dispersed Phase Velocity [*{of}* *SpeciesName*] = User_Function ([Name_X = *name_x*] | [Name_Y = *name_y*] | [Name_Z = *name_z*] | X = *x* | [Multiplier = *multiplier*])

Scope

Aria Material

Summary

Value from a Vector User Function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name_x	"string"	–
name_y	"string"	–
name_z	"string"	–
x	"string"	–
multiplier	real	1

User_Plugin

Syntax

Dispersed Phase Velocity [*{of}* *SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	“string”	–
plugin_parameters	“string”	–

Vector_String_Function

Syntax

Dispersed Phase Velocity [*{of}* *SpeciesName*] = Vector_String_Function (F_X = *f_x* | F_Y = *f_y* | [F_Z = *f_z*])

Scope

Aria Material

Summary

Value from a string function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f_x	“string”	–
f_y	“string”	–
f_z	“string”	–

Dispersed Phase Volume Fraction

Constant

Syntax

Dispersed Phase Volume Fraction [*{of}* *SpeciesName*] = Constant Value = *value*

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
value	real	–

Copied

Syntax

Dispersed Phase Volume Fraction [*{of}* *SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	“string”	–

Exponential

Syntax

Dispersed Phase Volume Fraction [*{of}* *SpeciesName*] = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Scope

Aria Material

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Dispersed Phase Volume Fraction [*{of}* *SpeciesName*] = Global (Global_Name = *global_name* | [Component = *component*])

Scope

Aria Material

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Dispersed Phase Volume Fraction [*{of}* *SpeciesName*] = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Dispersed Phase Volume Fraction [*{of} SpeciesName*] = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Dispersed Phase Volume Fraction [*{of} SpeciesName*] = Scalar_String_Function F = *f*

Scope

Aria Material

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	"string"	–

User_Field

Syntax

Dispersed Phase Volume Fraction [*{of} SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Dispersed Phase Volume Fraction [*{of} SpeciesName*] = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Scope

Aria Material

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Dispersed Phase Volume Fraction [*{of}* *SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	“string”	–
plugin_parameters	“string”	–

Distribution Coefficient

Competitive_Langmuir

Syntax

Distribution Coefficient [*{of}* *SpeciesName*] = Competitive_Langmuir (K_I = *k_i* | K_J = *k_j* | K_Ie = *k_ie* | Species_J = *species_j*)

Scope

Aria Material

Summary

This model has been deprecated and removed from the code.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
k_i	real	–
k_j	real	–
k_ie	real	–
species_j	integer	–

Constant

Syntax

Distribution Coefficient [*{of} SpeciesName*] = Constant *{kd | value}* = *kd*

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
kd	real	–

Copied

Syntax

Distribution Coefficient [*{of} SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	“string”	–

Exponential

Syntax

Distribution Coefficient [*{of}* *SpeciesName*] = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Scope

Aria Material

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Distribution Coefficient [*{of}* *SpeciesName*] = Global (Global_Name = *global_name* | [Component = *component*])

Scope

Aria Material

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
global_name	"string"	–
component	integer	0

Langmuir

Syntax

Distribution Coefficient [*{of}* *SpeciesName*] = Langmuir (K = *k* | K_I = *k_I*)

Scope

Aria Material

Summary

This model has been deprecated and removed from the code.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
k	real	–
k_I	real	–

Nd_Table

Syntax

Distribution Coefficient [*{of}* *SpeciesName*] = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Distribution Coefficient [*{of}* *SpeciesName*] = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Distribution Coefficient [*{of}* *SpeciesName*] = Scalar_String_Function F = *f*

Scope

Aria Material

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be

a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	"string"	–

User_Field

Syntax

Distribution Coefficient [*{of}* *SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Distribution Coefficient [*{of}* *SpeciesName*] = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Scope

Aria Material

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The

meaning of each option is as follows:

Toggle
If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1
toggle	"string"	–

User_Plugin

Syntax
Distribution Coefficient [*{of} SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope
Aria Material

Summary
Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	"string"	–
plugin_parameters	"string"	–

E

Effective Diffusivity

Hdiff

Syntax
Effective Diffusivity [*{of} SpeciesName*] = Hdiff (K_T = *k_t* | N_L = *n_l*)

Scope

Aria Material

Summary

Effective Hydrogen Diffusivity $D = 1 + \frac{N_T N_L}{(c_L \theta_T)^2}$

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
k_t	real	–
n_l	real	–

Electric Displacement

Basic

Syntax

Electric Displacement [*{of}* *SpeciesName*] = Basic ()

Scope

Aria Material

Summary

Basic electric displacement

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–

Generalized

Syntax

Electric Displacement [*{of}* *SpeciesName*] = Generalized ()

Scope

Aria Material

Summary

Generalized electric displacement

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–

Linear

Syntax

Electric Displacement [*{of}* *SpeciesName*] = Linear ()

Scope

Aria Material

Summary

Linear electric displacement

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–

Electrical Conductivity

Arrhenius

Syntax

Electrical Conductivity [*{of}* *SpeciesName*] = Arrhenius (R = *r* | [Sigma0 = *sigma0*] | [A = *a*] | Ea = *Ea*)

Scope

Aria Material

Summary

Arrhenius electrical conductivity, $\sigma = \sigma_0 + A * exp\left(-\frac{E_a}{RT}\right)$

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
r	real	–
sigma0	real	0
a	real	1
Ea	real	–

Bruggeman_Volume_Averaged

Syntax

Electrical Conductivity [*{of} SpeciesName*] = Bruggeman_Volume_Averaged [Exponent = *exponent*]

Scope

Aria Material

Summary

This model averages the properties of each species in the same material phase based on their volume fraction assuming a Bruggeman tortuosity model.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
exponent	real	1.5

Constant

Syntax

Electrical Conductivity [*{of} SpeciesName*] = Constant *{sigma | value}* = *sigma*

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
sigma	real	–

Copied

Syntax

Electrical Conductivity [*{of} SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	"string"	–

Electrode

Syntax

Electrical Conductivity [*{of}* *SpeciesName*] = Electrode ()

Scope

Aria Material

Summary

Deprecated model. Replace with string function or polynomial generic model.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–

Exponential

Syntax

Electrical Conductivity [*{of}* *SpeciesName*] = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Scope

Aria Material

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

From_Resistivity

Syntax

Electrical Conductivity [*{of}* *SpeciesName*] = From_Resistivity ()

Scope

Aria Material

Summary

Electrical conductivity calculated from the electrical resistance

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–

Global

Syntax

Electrical Conductivity [*{of}* *SpeciesName*] = Global (Global_Name = *global_name* | [Component = *component*])

Scope

Aria Material

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Electrical Conductivity [*{of}* *SpeciesName*] = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Electrical Conductivity [*{of}* *SpeciesName*] = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Electrical Conductivity [*{of} SpeciesName*] = Scalar_String_Function F = *f*

Scope

Aria Material

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	"string"	–

Tbc

Syntax

Electrical Conductivity [*{of}* *SpeciesName*] = Tbc (Ki = *ki* | Ti = *ti* | E = *e* | R = *r*)

Scope

Aria Material

Summary

This model has been deprecated and removed from the code.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
ki	real	–
ti	real	–
e	real	–
r	real	–

Thermal

Syntax

Electrical Conductivity [*{of}* *SpeciesName*] = Thermal ([A = *a*] | [B = *b*] | [C = *c*] | [D = *d*])

Scope

Aria Material

Summary

This model has been deprecated and removed from the code. Use the generic polynomial model.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
a	real	0
b	real	0
c	real	0
d	real	0

User_Field

Syntax

Electrical Conductivity [*{of}* *SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Electrical Conductivity [*{of}* *SpeciesName*] = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Scope

Aria Material

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Electrical Conductivity [*{of} SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	“string”	–
plugin_parameters	“string”	–

Electrical Permittivity

Constant

Syntax

Electrical Permittivity [*{of} SpeciesName*] = Constant *{kappa | value}* = *kappa*

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
kappa	real	–

Copied

Syntax

Electrical Permittivity [*{of}* *SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	“string”	–

Exponential

Syntax

Electrical Permittivity [*{of}* *SpeciesName*] = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Scope

Aria Material

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Electrical Permittivity [*{of} SpeciesName*] = Global (Global_Name = *global_name* | [Component = *component*])

Scope

Aria Material

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Electrical Permittivity [*{of} SpeciesName*] = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Electrical Permittivity [*{of} SpeciesName*] = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Electrical Permittivity [*{of}* *SpeciesName*] = Scalar_String_Function F = *f*

Scope

Aria Material

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	"string"	–

User_Field

Syntax

Electrical Permittivity [*{of}* *SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Electrical Permittivity [*{of}* *SpeciesName*] = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Scope

Aria Material

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Electrical Permittivity [*{of} SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	“string”	–
plugin_parameters	“string”	–

Electrical Resistance

Constant

Syntax

Electrical Resistance [*{of} SpeciesName*] = Constant Value = *value*

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
value	real	–

Copied

Syntax

Electrical Resistance [*{of}* *SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	“string”	–

Exponential

Syntax

Electrical Resistance [*{of}* *SpeciesName*] = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Scope

Aria Material

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Electrical Resistance [*{of} SpeciesName*] = Global (Global_Name = *global_name* | [Component = *component*])

Scope

Aria Material

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Electrical Resistance [*{of} SpeciesName*] = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Electrical Resistance [*{of}* *SpeciesName*] = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Electrical Resistance [*{of} SpeciesName*] = Scalar_String_Function F = *f*

Scope

Aria Material

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	"string"	–

User_Field

Syntax

Electrical Resistance [*{of}* *SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Electrical Resistance [*{of}* *SpeciesName*] = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Scope

Aria Material

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Electrical Resistance [*{of} SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	“string”	–
plugin_parameters	“string”	–

Electrical Resistivity

Constant

Syntax

Electrical Resistivity [*{of} SpeciesName*] = Constant *{rho | value}* = *rho*

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
rho	real	–

Copied

Syntax

Electrical Resistivity [*{of}* *SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	“string”	–

Exponential

Syntax

Electrical Resistivity [*{of}* *SpeciesName*] = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Scope

Aria Material

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

From_Conductivity

Syntax

Electrical Resistivity [*{of} SpeciesName*] = From_Conductivity ()

Scope

Aria Material

Summary

Electrical resistivity calculated from the electrical conductivity

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–

Global

Syntax

Electrical Resistivity [*{of} SpeciesName*] = Global (Global_Name = *global_name* | [Component = *component*])

Scope

Aria Material

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Electrical Resistivity [*{of} SpeciesName*] = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Electrical Resistivity [*{of} SpeciesName*] = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Electrical Resistivity [*{of}* *SpeciesName*] = Scalar_String_Function F = *f*

Scope

Aria Material

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	"string"	–

User_Field

Syntax

Electrical Resistivity [*{of}* *SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Electrical Resistivity [*{of}* *SpeciesName*] = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Scope

Aria Material

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Electrical Resistivity [*{of}* *SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	“string”	–
plugin_parameters	“string”	–

Electrode Reaction Details

Electrode_Object

Syntax

Electrode Reaction Details [*{of}* *SpeciesName*] = Electrode_Object Reaction_Index = *reaction_index*

Scope

Aria Material

Summary

Reaction details (overpotential and current) for a specific electrode reaction

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
reaction_index	integer	–

Electromigration Coefficient

Constant

Syntax

Electromigration Coefficient [*{of}* *SpeciesName*] = Constant *{m | value}* = *m*

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
m	real	–

Copied

Syntax

Electromigration Coefficient [*{of}* *SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	“string”	–

Exponential

Syntax

Electromigration Coefficient [*{of}* *SpeciesName*] = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Scope

Aria Material

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Electromigration Coefficient [*{of}* *SpeciesName*] = Global (Global_Name = *global_name* | [Component = *component*])

Scope

Aria Material

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Electromigration Coefficient [*{of}* *SpeciesName*] = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Electromigration Coefficient [*{of}* *SpeciesName*] = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Electromigration Coefficient [*{of} SpeciesName*] = Scalar_String_Function F = *f*

Scope

Aria Material

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	"string"	–

User_Field

Syntax

Electromigration Coefficient [*{of}* *SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Electromigration Coefficient [*{of}* *SpeciesName*] = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Scope

Aria Material

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Electromigration Coefficient [*{of} SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	“string”	–
plugin_parameters	“string”	–

Emissivity

Banded_Wavelength

Syntax

Emissivity [*{of} SpeciesName*] = Banded_Wavelength (E0 = *e0* | E1 = *e1* | [E2 = *e2*] | [E3 = *e3*] | [E4 = *e4*] | [E5 = *e5*] | [E6 = *e6*] | [E7 = *e7*] | Wavelength_Band_Model = *wavelength_band_model*)

Scope

Aria Material

Summary

Banded wavelength emissivity.

Description

Computes emissivity as a sum over wavelength bands. This requires a 'banded wavelength model' block defined elsewhere which defines the band wavelengths and is referenced here using the 'WAVELENGTH_BAND_MODEL' argument. Emissivities of each band in the selected model (ε_i) must be provided as 'E0', 'E1', etc. . .

$$\varepsilon = \sum_i F_i \varepsilon_i$$

where the band fraction, F_i is calculated from the band start and end wavelengths ($\lambda_{i,s}$ and $\lambda_{i,e}$) black body fraction (B) as

$$F_i = B(\lambda_{i,e} T) - B(\lambda_{i,s})$$

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
e0	real	–
e1	real	–
e2	real	0
e3	real	0
e4	real	0
e5	real	0
e6	real	0
e7	real	0
wavelength_band_model	"string"	–

Calore_User_Sub

Syntax

Emissivity [*{of}* *SpeciesName*] = Calore_User_Sub (Name = *name* | Type = *type* | [Multiplier = *multiplier*] | [Material_Data_Block = *material_data_block*] | [Data = *data*] | [Scaling_Field = *scaling_field*])

Scope

Aria Material

Summary

Values from a Calore user subroutine

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
type	“string”	–
multiplier	real	1
material_data_block	“string”	–
data	“string”	–
scaling_field	“string”	–

Constant

Syntax

Emissivity [*{of}* *SpeciesName*] = Constant *{e | value}* = *e*

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
e	real	–

Copied

Syntax

Emissivity [*{of}* *SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	“string”	–

Exponential

Syntax

Emissivity [*{of}* *SpeciesName*] = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Scope

Aria Material

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Fortran

Syntax

Emissivity [*{of}* *SpeciesName*] = Fortran ([Multiplier = *multiplier*] | Sub_Name = *sub_name* | [Real_Data = *real_data*] | [Int_Data = *int_data*] | [Resource_Name = *resource_name*] | [Data = *data*])

Scope

Aria Material

Summary

Fortran subroutine emissivity.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
multiplier	real	1
sub_name	“string”	–
real_data	“string”	–
int_data	“string”	–
resource_name	“string”	–
data	“string”	–

Global

Syntax

Emissivity [*{of}* *SpeciesName*] = Global (Global_Name = *global_name* | [Component = *component*])

Scope

Aria Material

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
global_name	“string”	–
component	integer	0

Mass_Average

Syntax

Emissivity [*{of}* *SpeciesName*] = Mass_Average ()

Scope

Aria Material

Summary

Emissivity computed from mass average of species emissivities

Description

Defines a mixture emissivity from mass fractions as

$$\epsilon = \frac{\sum_i Y_i \epsilon_i}{\sum_i Y_i}$$

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–

Nd_Table

Syntax

Emissivity [*{of} SpeciesName*] = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Emissivity [*{of} SpeciesName*] = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Emissivity [*{of} SpeciesName*] = Scalar_String_Function F = *f*

Scope

Aria Material

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	"string"	–

User_Field

Syntax

Emissivity [*{of}* *SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Emissivity [*{of}* *SpeciesName*] = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Scope

Aria Material

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Emissivity [*{of}* *SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	“string”	–
plugin_parameters	“string”	–

Volume_Average

Syntax

Emissivity [*{of}* *SpeciesName*] = Volume_Average ()

Scope

Aria Material

Summary

Emissivity computed from volume average of species emissivities

Description

Defines a mixture emissivity from volume fractions as

$$\epsilon = \frac{\sum_i V_{f,i}\epsilon_i}{\sum_i V_{f,i}}$$

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–

Enclosure Mbk

Calore_User_Sub

Syntax

Enclosure Mbk [*{of}* *SpeciesName*] = Calore_User_Sub (Name = *name* | Type = *type* | [Multiplier = *multiplier*] | [Material_Data_Block = *material_data_block*] | [Data = *data*] | [Scaling_Field = *scaling_field*])

Scope

Aria Material

Summary

Values from a Calore user subroutine

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
type	“string”	–
multiplier	real	1
material_data_block	“string”	–
data	“string”	–
scaling_field	“string”	–

Constant

Syntax

Enclosure Mbk [*{of}* *SpeciesName*] = Constant *{k | value}* = *k*

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
k	real	–

Copied

Syntax

Enclosure Mbk [*{of}* *SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	“string”	–

Exponential

Syntax

Enclosure Mbk [*{of}* *SpeciesName*] = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Scope

Aria Material

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Enclosure Mbk [*{of} SpeciesName*] = Global (Global_Name = *global_name* | [Component = *component*])

Scope

Aria Material

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Enclosure Mbk [*{of} SpeciesName*] = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Enclosure Mbk [*{of}* *SpeciesName*] = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Enclosure Mbk [*{of} SpeciesName*] = Scalar_String_Function F = *f*

Scope

Aria Material

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	"string"	–

User_Field

Syntax

Enclosure Mbk [*{of}* *SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Enclosure Mbk [*{of}* *SpeciesName*] = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Scope

Aria Material

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Enclosure Mbk [*{of}* *SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	“string”	–
plugin_parameters	“string”	–

Enclosure Mbl

Calore_User_Sub

Syntax

Enclosure Mbl [*{of}* *SpeciesName*] = Calore_User_Sub (Name = *name* | Type = *type* | [Multiplier = *multiplier*] | [Material_Data_Block = *material_data_block*] | [Data = *data*] | [Scaling_Field = *scaling_field*])

Scope

Aria Material

Summary

Values from a Calore user subroutine

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
type	“string”	–
multiplier	real	1
material_data_block	“string”	–
data	“string”	–
scaling_field	“string”	–

Constant

Syntax

Enclosure Mbl [*{of} SpeciesName*] = Constant *{l | value}* = *l*

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
l	real	–

Copied

Syntax

Enclosure Mbl [*{of} SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	“string”	–

Exponential

Syntax

Enclosure Mbl [*{of}* *SpeciesName*] = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Scope

Aria Material

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

From_Geometry

Syntax

Enclosure Mbl [*{of}* *SpeciesName*] = From_Geometry ([Prefactor = *prefactor*] | Enclosure = *enclosure*)

Scope

Aria Material

Summary

Values from geometry with a user selectable pre-factor

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
prefactor	real	3.6
enclosure	"string"	–

Global

Syntax

Enclosure Mbl [*{of} SpeciesName*] = Global (Global_Name = *global_name* | [Component = *component*])

Scope

Aria Material

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Enclosure Mbl [*{of} SpeciesName*] = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
vars	"string"1["string"2["string"3["string"4["string"5]]]]	–
i0	"string"	–
i1	"string"	–
i2	"string"	–
x	"string"	–
y	"string"	–
z	"string"	–

Polynomial

Syntax

Enclosure Mbl [*{of}* *SpeciesName*] = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Enclosure Mbl [*{of}* *SpeciesName*] = Scalar_String_Function F = *f*

Scope

Aria Material

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be

a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	"string"	–

User_Field

Syntax

Enclosure Mbl [*{of} SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Enclosure Mbl [*{of} SpeciesName*] = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Scope

Aria Material

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Enclosure Mbl [*{of} SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	“string”	–
plugin_parameters	“string”	–

Energy Diffusive Flux

Basic

Syntax

Energy Diffusive Flux [*{of} SpeciesName*] = Basic ()

Scope

Aria Material

Summary

Basic model of energy diffusive flux

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–

Energy_Mass

Syntax

Energy Diffusive Flux [*{of}* *SpeciesName*] = Energy_Mass ()

Scope

Aria Material

Summary

Energy mass model of energy diffusive flux

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–

From_Mixture_Mass_Diffusivity

Syntax

Energy Diffusive Flux [*{of}* *SpeciesName*] = From_Mixture_Mass_Diffusivity ()

Scope

Aria Material

Summary

Energy diffusive flux = -density * mixture_mass_diffusivity * grad(enthalpy)

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–

Energy Face Stabilization Scaling

Default

Syntax

Energy Face Stabilization Scaling ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) = Default ()

Scope

Aria Material

Summary

Default face stabilization scaling for energy equation

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–

Enthalpy

Cantera

Syntax

Enthalpy [*{of}* *SpeciesName*] = Cantera [Poffset = *poffset*]

Scope

Aria Material

Summary

Cantera enthalpy

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
poffset	real	0

Constant

Syntax

Enthalpy [*{of}* *SpeciesName*] = Constant *{h | value}* = *h*

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
h	real	–

Copied

Syntax

Enthalpy [*{of}* *SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	“string”	–

Dirichlet_From_Temperature

Syntax

Enthalpy [*{of}* *SpeciesName*] = Dirichlet_From_Temperature ([Temperature = *temperature*] | [Pressure = *pressure*] | [Mass_Fraction_0 = *mass_fraction_0*] | [Mass_Fraction_1 = *mass_fraction_1*] | [Mass_Fraction_2 = *mass_fraction_2*] | [Mass_Fraction_3 = *mass_fraction_3*] | [Mass_Fraction_4 = *mass_fraction_4*] | [Mass_Fraction_5 = *mass_fraction_5*] | [Mass_Fraction_6 = *mass_fraction_6*] | [Mass_Fraction_7 = *mass_fraction_7*] | [Mass_Fraction_8 = *mass_fraction_8*])

Scope

Aria Material

Summary

Enthalpy computed from mass fraction, temperature, and pressure

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
temperature	real	0
pressure	real	0
mass_fraction_0	real	0
mass_fraction_1	real	0
mass_fraction_2	real	0
mass_fraction_3	real	0
mass_fraction_4	real	0
mass_fraction_5	real	0
mass_fraction_6	real	0
mass_fraction_7	real	0
mass_fraction_8	real	0

Exponential

Syntax

Enthalpy [*{of}* *SpeciesName*] = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Scope

Aria Material

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	“string”	–
constant	real	0
multiplier	real	1
exponent	real	–

From_Temperature

Syntax

Enthalpy [*{of}* *SpeciesName*] = From_Temperature (*{t | value}* = *t* | [Poffset = *poffset*])

Scope

Aria Material

Summary

Cantera enthalpy computed from temperature

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
t	real	–
poffset	real	0

Global

Syntax

Enthalpy [*{of}* *SpeciesName*] = Global (Global_Name = *global_name* | [Component = *component*])

Scope

Aria Material

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
global_name	“string”	–
component	integer	0

Mass_Average

Syntax

Enthalpy [*{of}* *SpeciesName*] = Mass_Average ()

Scope

Aria Material

Summary

Enthalpy computed from mass average of species enthalpies

Description

Defines a mixture enthalpy from mass fractions as

$$h = \frac{\sum_i Y_i h_i}{\sum_i Y_i}$$

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–

Nd_Table

Syntax

Enthalpy [*{of}* *SpeciesName*] = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Enthalpy [*{of}* *SpeciesName*] = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Enthalpy [*{of} SpeciesName*] = Scalar_String_Function F = *f*

Scope

Aria Material

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	"string"	–

User_Field

Syntax

Enthalpy [*{of} SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Enthalpy [*{of} SpeciesName*] = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Scope

Aria Material

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Enthalpy [*{of}* *SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	“string”	–
plugin_parameters	“string”	–

Enthalpy Advection

Material_Phase_Sum

Syntax

Enthalpy Advection [*{of}* *SpeciesName*] = Material_Phase_Sum ()

Scope

Aria Material

Summary

Sum the enthalpy advection terms from multiple material phases for use in a one temperature model.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–

Porous

Syntax

Enthalpy Advection [*{of}* *SpeciesName*] = Porous ()

Scope

Aria Material

Summary

Provides the advective term for porous enthalpy equation.

Description

For species i, phase p, this model provides the advective term for the porous enthalpy equation,

$$\mathbf{m}_p h_{i,p},$$

where $h_{i,p}$ is the species enthalpy and $\mathbf{m}_p$ is the mass flux for the given phase. The phase mass flux is also provided by this model assuming it was not already provided elsewhere. By default, the non-relative mass flux is provided,

$$\mathbf{m}_p = \rho_p \mathbf{u}_p,$$

which is applicable to problems with or without mesh motion since we automatically include an appropriate volumetric mesh motion correction term to the porous enthalpy equation.

If the `use relative mass flux` porous flow option is active, the phase mass flux will be relative to the mesh. For phases other than the solid phase, the relative mass flux is

$$\mathbf{m}_p = \rho_p \left(\mathbf{u}_p - \phi S_p \hat{\mathbf{u}} \right),$$

where ρ_p is the density of the phase, ϕ is the porosity, S_p is the optional phase saturation, $\hat{\mathbf{u}}$ is the mesh velocity, and $\mathbf{u}_p$ is the phase velocity (e.g., Darcy velocity model). Note if an optional phase saturation is provided, the pore volume is occupied by multiple phases in accordance to their saturations, otherwise the pore volume is fully occupied by a single phase.

For the solid phase, the relative mass flux is

$$\mathbf{m}_{i,s} = \rho_{i,s} \left(\mathbf{u}_s - \hat{\mathbf{u}} \right).$$

For problems involving deformation, the Darcy velocity should be specified relative to the deforming matrix velocity e.g. solid phase. Lastly, if there is no mesh motion, the mesh motion flux will not be included.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–

Porous_Muscl

Syntax

Enthalpy Advection [*{of}* *SpeciesName*] = Porous_Muscl ([Limiter = *limiter*] | [Zeta = *zeta*] | [Use_Peclet_Blending = *use_peclet_blending*])

Scope

Aria Material

Summary

MUSCL scheme for advective flux in porous equations. Currently only valid for EQ porous_enthalpy / mass_balance while using CVFEM.

Description

Let q be the subcontrol surface interpolated scalar and let q_{muscl} be the MUSCL scalar at the subcontrol surface computed using the projected nodal gradient of the scalar and the prescribed limiter (i.e., `NONE`, `SUPERBEE`, `MINDMOD`, `VAN_ALBADA`, `VAN_LEER`). The scalar at the subcontrol surface is linearly interpolated as

$$q_f = \chi q_{muscl} + (1 - \chi)\, q,$$

where $\chi = \dfrac{(\zeta Pe)^2}{5 + (\zeta Pe)^2}$ is the weighting function between 0 and 1 that is based on the cell Peclet number Pe and ζ, a user specified scaling. If no diffusion is present, (i.e., pure advection), q_{muscl} is used. Similarly, if the cell Peclet number is near zero, then the interpolated scalar q is used.

The cell Peclet number is the ratio of advective flux to diffusive flux in a cell. For the mass balance equation, it is assumed the `MASS_BALANCE_DIFFUSIVE_FLUX = POROUS` is used, giving a diffusive flux (if present) of the form $\mathbf{J} = -\phi\rho D_i \nabla Y_i$, which in turn yields the following cell Pe

$$Pe_{\Delta x} = \frac{\mathbf{u} \cdot \mathbf{\Delta x}}{\phi D_i}.$$

On the other hand, for the gas phase porous enthalpy equation, it is assumed the `HEAT CONDUCTION = POROUS_SIMPLIFIED_DIFFUSIVE_ENTHALPY` model is

used, giving a diffusive flux (if present) of the form $\mathbf{q} = -\phi\rho D\nabla h$, which in turn for a unit Lewis number yields the following cell Pe,

$$Pe_{\Delta x} = \frac{\mathbf{u}\cdot\mathbf{\Delta x}}{\phi\alpha}$$

For the solid phase (or no phase) porous enthalpy equation, it is assumed `HEAT CONDUCTION = FOURIERS_LAW` is used i.e, $\mathbf{q} = -k\nabla T$, resulting in the following cell Pe

$$Pe_{\Delta x} = \frac{\mathbf{u}\cdot\mathbf{\Delta x}}{\alpha}.$$

Note the thermal diffusivity is computed using a scalar thermal conductivity. For other cases, set the optional parameter `use_pecleting_blending = 0`, to turn off Peclet blending.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
limiter	"string"	VAN_LEER
zeta	real	1
use_peclet_blending	integer	–

Porous_Upwind

Syntax

Enthalpy Advection [*{of} SpeciesName*] = Porous_Upwind ()

Scope

Aria Material

Summary

Upwinded advected quantity for porous enthalpy equation. Only valid for CVFEM.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–

Entrance Distance

Correlation_Bar

Syntax

Entrance Distance [*{of}* *SpeciesName*] = Correlation_Bar [Bulk_Node_Interface = *bulk_node_interface*]

Scope

Aria Material

Summary

Correction factor for difference between near-wall and bulk fluid state for Gnielinski pipe and annulus correlations.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
bulk_node_interface	"string"	FALSE

Entrance Length

Correlation

Syntax

Entrance Length [*{of}* *SpeciesName*] = Correlation (De = *De* | [Bulk_Node_Interface = *bulk_node_interface*])

Scope

Aria Material

Summary

Correlation Entrance Length

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
De	real	–
bulk_node_interface	"string"	–

Entry Pressure

Constant

Syntax

Entry Pressure [*{of}* *SpeciesName*] = Constant *{pe | value}* = *pe*

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
pe	real	–

Copied

Syntax

Entry Pressure [*{of}* *SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	“string”	–

Exponential

Syntax

Entry Pressure [*{of}* *SpeciesName*] = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Scope

Aria Material

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Entry Pressure [*{of} SpeciesName*] = Global (Global_Name = *global_name* | [Component = *component*])

Scope

Aria Material

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Entry Pressure [*{of} SpeciesName*] = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Entry Pressure [*{of} SpeciesName*] = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Entry Pressure [*{of} SpeciesName*] = Scalar_String_Function F = *f*

Scope

Aria Material

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	"string"	–

User_Field

Syntax

Entry Pressure [*{of}* *SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Entry Pressure [*{of}* *SpeciesName*] = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Scope

Aria Material

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Entry Pressure [*{of}* *SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	“string”	–
plugin_parameters	“string”	–

Equilibrium Constant

Arrhenius

Syntax

Equilibrium Constant [*{of}* *SpeciesName*] = Arrhenius ([C = *c*] | [M = *m*] | W_B = *w_b* | R = *r*)

Scope

Aria Material

Summary

Arrhenius Equilibrium Constant, $E = C + M * exp\left(-\frac{W_B}{RT}\right)$

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
c	real	0
m	real	1
w_b	real	–
r	real	–

Equilibrium Potential

Constant

Syntax

Equilibrium Potential [*{of}* *SpeciesName*] = Constant *{u | value}* = *u*

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
u	real	–

Copied

Syntax

Equilibrium Potential [*{of}* *SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	“string”	–

Exponential

Syntax

Equilibrium Potential [*{of}* *SpeciesName*] = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Scope

Aria Material

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

From_Chemical_Potential

Syntax

Equilibrium Potential [*{of}* *SpeciesName*] = From_Chemical_Potential ([F = *f*] | Species_Name = *species_name* | [Species_Phase = *species_phase*])

Scope

Aria Material

Summary

Determine the equilibrium voltage based on the chemical potential of a selected species where $\phi_{equil} = -\mu/F$.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	real	9.649e+04
species_name	"string"	–
species_phase	"string"	–

Global

Syntax

Equilibrium Potential [*{of}* *SpeciesName*] = Global (Global_Name = *global_name* | [Component = *component*])

Scope

Aria Material

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
global_name	“string”	–
component	integer	0

Lcoo2

Syntax

Equilibrium Potential [*{of}* *SpeciesName*] = Lcoo2 ()

Scope

Aria Material

Summary

Equilibrium potential

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–

Nd_Table

Syntax

Equilibrium Potential [*{of}* *SpeciesName*] = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Nernst

Syntax

Equilibrium Potential [*{of}* *SpeciesName*] = Nernst (Species_A = *species_a* | Species_B = *species_b* | [F = *f*] | [R = *r*] | [Eps = *eps*] | U = *u*)

Scope

Aria Material

Summary

Nernst-Einstein form of the equilibrium potential

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
species_a	“string”	–
species_b	“string”	–
f	real	9.649e+04
r	real	8.314
eps	real	0.001
u	real	–

Polynomial

Syntax

Equilibrium Potential [*{of}* *SpeciesName*] = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Equilibrium Potential [*{of}* *SpeciesName*] = Scalar_String_Function F = *f*

Scope

Aria Material

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be

a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	"string"	–

Two_Phase

Syntax

Equilibrium Potential [*{of}* *SpeciesName*] = Two_Phase (Cmin = *Cmin* | Cmax = *Cmax* | [R = *r*] | Species_Name = *species_name*)

Scope

Aria Material

Summary

Equilibrium potential that is constant between Cmin and Cmax corresponding to a two-phase region. Outside of that range it behaves as an ideal solution.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Cmin	real	–
Cmax	real	–
r	real	8.314
species_name	"string"	–

User_Field

Syntax

Equilibrium Potential [*{of}* *SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Equilibrium Potential [*{of}* *SpeciesName*] = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Scope

Aria Material

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1
toggle	"string"	–

User_Plugin

Syntax

Equilibrium Potential [*{of} SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	“string”	–
plugin_parameters	“string”	–

Equilibrium Pressure

Constant

Syntax

Equilibrium Pressure [*{of} SpeciesName*] = Constant *{p | value}* = *p*

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
p	real	–

Copied

Syntax

Equilibrium Pressure [*{of}* *SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	"string"	–

Exponential

Syntax

Equilibrium Pressure [*{of}* *SpeciesName*] = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Scope

Aria Material

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Equilibrium Pressure [*{of}* *SpeciesName*] = Global (Global_Name = *global_name* | [Component = *component*])

Scope

Aria Material

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
global_name	“string”	–
component	integer	0

Nd_Table

Syntax

Equilibrium Pressure [*{of}* *SpeciesName*] = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Equilibrium Pressure [*{of}* *SpeciesName*] = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Equilibrium Pressure [*{of}* *SpeciesName*] = Scalar_String_Function F = *f*

Scope

Aria Material

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be

a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	"string"	–

User_Field

Syntax

Equilibrium Pressure [*{of}* *SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Equilibrium Pressure [*{of}* *SpeciesName*] = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Scope

Aria Material

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1
toggle	"string"	–

User_Plugin

Syntax

Equilibrium Pressure [*{of} SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	"string"	–
plugin_parameters	"string"	–

Exchange Current Density

Constant

Syntax

Exchange Current Density [*{of} SpeciesName*] = Constant Value = *value*

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
value	real	–

Copied

Syntax

Exchange Current Density [*{of}* *SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	“string”	–

Exponential

Syntax

Exchange Current Density [*{of}* *SpeciesName*] = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Scope

Aria Material

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Exchange Current Density [*{of}* *SpeciesName*] = Global (Global_Name = *global_name* | [Component = *component*])

Scope

Aria Material

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Exchange Current Density [*{of}* *SpeciesName*] = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Exchange Current Density [*{of}* *SpeciesName*] = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Rate_Constant_Species

Syntax

Exchange Current Density [*{of}* *SpeciesName*] = Rate_Constant_Species ([F = *f*] | K = *k* | [Liquid_Exponent = *liquid_exponent*] | [Solid_Exponent = *solid_exponent*] | [Cs_Max = *Cs_max*] | Liquid_Species = *liquid_species* | Solid_Species = *solid_species* | [Reaction_Name = *reaction_name*])

Scope

Aria Material

Summary

Computes an exchange current density of the form
$i_0 = kF \prod \text{liquid_species}^{\text{liquid_exponent}} \times \prod \text{solid_species}^{\text{solid_exponent}}$
where the species whose concentrations are included are user-specified parameters.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	real	9.649e+04
k	real	–
liquid_exponent	real	0.5
solid_exponent	real	0.5
Cs_max	real	-1
liquid_species	"string"	–
solid_species	"string"	–
reaction_name	"string"	–

Scalar_String_Function

Syntax

Exchange Current Density [*{of}* *SpeciesName*] = Scalar_String_Function F = *f*

Scope

Aria Material

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	"string"	–

User_Field

Syntax

Exchange Current Density [*{of}* *SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Exchange Current Density [*{of}* *SpeciesName*] = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Scope

Aria Material

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1
toggle	"string"	–

User_Plugin

Syntax

Exchange Current Density [*{of}* *SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	“string”	–
plugin_parameters	“string”	–

Extension Speed

Constant

Syntax

Extension Speed [*{of}* *SpeciesName*] = Constant Value = *value*

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
value	real	–

Copied

Syntax

Extension Speed [*{of}* *SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	"string"	–

Exponential

Syntax

Extension Speed [*{of}* *SpeciesName*] = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Scope

Aria Material

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Extension Speed [*{of}* *SpeciesName*] = Global (Global_Name = *global_name* | [Component = *component*])

Scope

Aria Material

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
global_name	“string”	–
component	integer	0

Nd_Table

Syntax

Extension Speed [*{of}* *SpeciesName*] = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Extension Speed [*{of}* *SpeciesName*] = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Extension Speed [*{of}* *SpeciesName*] = Scalar_String_Function F = *f*

Scope

Aria Material

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be

a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	"string"	–

User_Field

Syntax

Extension Speed [*{of}* *SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Extension Speed [*{of}* *SpeciesName*] = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Scope

Aria Material

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1
toggle	"string"	–

User_Plugin

Syntax

Extension Speed [*{of}* *SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	"string"	–
plugin_parameters	"string"	–

Extension Speed Diffusive Flux

Definition

Syntax

Extension Speed Diffusive Flux [*{of}* *SpeciesName*] = Definition ()

Scope

Aria Material

Summary

The definition of the extension speed diffusive flux

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–

Extension Speed Multiplier

Constant

Syntax

Extension Speed Multiplier [*{of}* *SpeciesName*] = Constant Value = *value*

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
value	real	–

Copied

Syntax

Extension Speed Multiplier [*{of}* *SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	“string”	–

Exponential

Syntax

Extension Speed Multiplier [*{of}* *SpeciesName*] = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Scope

Aria Material

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Extension Speed Multiplier [*{of}* *SpeciesName*] = Global (Global_Name = *global_name* | [Component = *component*])

Scope

Aria Material

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Extension Speed Multiplier [*{of}* *SpeciesName*] = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Extension Speed Multiplier [*{of}* *SpeciesName*] = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Extension Speed Multiplier [*{of}* *SpeciesName*] = Scalar_String_Function F = *f*

Scope

Aria Material

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	"string"	–

User_Field

Syntax

Extension Speed Multiplier [*{of}* *SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Extension Speed Multiplier [*{of}* *SpeciesName*] = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Scope

Aria Material

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Extension Speed Multiplier [*{of} SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	“string”	–
plugin_parameters	“string”	–

F

Fission Cross Section

Constant

Syntax

Fission Cross Section [*{of} SpeciesName*] = Constant *{fiss | value}* = *fiss*

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
fiss	real	–

Copied

Syntax

Fission Cross Section [*{of}* *SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	“string”	–

Exponential

Syntax

Fission Cross Section [*{of}* *SpeciesName*] = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Scope

Aria Material

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Fission Cross Section [*{of}* *SpeciesName*] = Global (Global_Name = *global_name* | [Component = *component*])

Scope

Aria Material

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
global_name	"string"	–
component	integer	0

Linearized

Syntax

Fission Cross Section [*{of}* *SpeciesName*] = Linearized (Sigma_0 = *sigma_0* | D_Sigma_D_Rho = *d_sigma_d_rho* | D_Sigma_D_T = *d_sigma_d_t* | T_Sigma_0 = *t_sigma_0* | Rho_Sigma_0 = *rho_sigma_0*)

Scope

Aria Material

Summary

Linearized fission cross section

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
sigma_0	real	–
d_sigma_d_rho	real	–
d_sigma_d_t	real	–
t_sigma_0	real	–
rho_sigma_0	real	–

Nd_Table

Syntax

Fission Cross Section [*{of}* *SpeciesName*] = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Fission Cross Section [*{of}* *SpeciesName*] = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Fission Cross Section [*{of}* *SpeciesName*] = Scalar_String_Function F = *f*

Scope

Aria Material

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be

a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	"string"	–

User_Field

Syntax

Fission Cross Section [*{of}* *SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Fission Cross Section [*{of}* *SpeciesName*] = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Scope

Aria Material

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Fission Cross Section [*{of} SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	“string”	–
plugin_parameters	“string”	–

Flow Area

Constant

Syntax

Flow Area [*{of} SpeciesName*] = Constant *{a | value}* = *a*

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
a	real	–

Copied

Syntax

Flow Area [*{of}* *SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	“string”	–

Correlation_Spatial_User_Function

Syntax

Flow Area [*{of}* *SpeciesName*] = Correlation_Spatial_User_Function ([Threshold = *threshold*] | [Multiplier = *multiplier*] | Name = *name* | Bar = *bar* | X = *x* | [Bulk_Node_Interface = *bulk_node_interface*])

Scope

Aria Material

Summary

Correlation spatially varying flow area based on a coordinate dependent user function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
threshold	real	-1.798e+308
multiplier	real	1
name	“string”	–
bar	“string”	–
x	“string”	–
bulk_node_interface	“string”	–

Exponential

Syntax

Flow Area [*{of}* *SpeciesName*] = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Scope

Aria Material

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	“string”	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Flow Area [*{of} SpeciesName*] = Global (Global_Name = *global_name* | [Component = *component*])

Scope

Aria Material

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
global_name	“string”	–
component	integer	0

Nd_Table

Syntax

Flow Area [*{of} SpeciesName*] = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Flow Area [*{of}* *SpeciesName*] = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Flow Area [*{of}* *SpeciesName*] = Scalar_String_Function F = *f*

Scope

Aria Material

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be

a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	"string"	–

Spatial_User_Function

Syntax

Flow Area [*{of}* *SpeciesName*] = Spatial_User_Function ([Threshold = *threshold*] | [Multiplier = *multiplier*] | Name = *name* | Bar = *bar* | X = *x*)

Scope

Aria Material

Summary

Spatially varying flow area based on a coordinate dependent user function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
threshold	real	-1.798e+308
multiplier	real	1
name	"string"	–
bar	"string"	–
x	"string"	–

User_Field

Syntax

Flow Area [*{of}* *SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Flow Area [*{of}* *SpeciesName*] = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Scope

Aria Material

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1
toggle	"string"	–

User_Plugin

Syntax

Flow Area [*{of} SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	“string”	–
plugin_parameters	“string”	–

Flowing Liquid Viscosity

Constant

Syntax

Flowing Liquid Viscosity [*{of} SpeciesName*] = Constant *{mu | value}* = *mu*

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
mu	real	–

Copied

Syntax

Flowing Liquid Viscosity [*{of}* *SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	"string"	–

Exponential

Syntax

Flowing Liquid Viscosity [*{of}* *SpeciesName*] = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Scope

Aria Material

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Flowing Liquid Viscosity [*{of}* *SpeciesName*] = Global (Global_Name = *global_name* | [Component = *component*])

Scope

Aria Material

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
global_name	“string”	–
component	integer	0

Nd_Table

Syntax

Flowing Liquid Viscosity [*{of}* *SpeciesName*] = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Flowing Liquid Viscosity [*{of}* *SpeciesName*] = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Flowing Liquid Viscosity [*{of}* *SpeciesName*] = Scalar_String_Function F = *f*

Scope

Aria Material

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be

a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	"string"	–

User_Field

Syntax

Flowing Liquid Viscosity [*{of}* *SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Flowing Liquid Viscosity [*{of}* *SpeciesName*] = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Scope

Aria Material

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1
toggle	"string"	–

User_Plugin

Syntax

Flowing Liquid Viscosity [*{of} SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	"string"	–
plugin_parameters	"string"	–

Fluid Beta

Correlation

Syntax

Fluid Beta [*{of} SpeciesName*] = Correlation ([Beta = *beta*] | [Bulk_Node_Interface = *bulk_node_interface*])

Scope

Aria Material

Summary

Correlation Expansion Coefficient

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
beta	real	0
bulk_node_interface	“string”	–

Fluid Density

Correlation

Syntax

Fluid Density [*{of}* *SpeciesName*] = Correlation ([Rho = *rho*] | [Bulk_Node_Interface = *bulk_node_interface*])

Scope

Aria Material

Summary

Correlation fluid temperature

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
rho	real	0
bulk_node_interface	“string”	–

Fluid Temperature

Constant

Syntax

Fluid Temperature [*{of}* *SpeciesName*] = Constant *{t | value}* = *t*

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
t	real	–

Copied

Syntax

Fluid Temperature [*{of}* *SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	“string”	–

Exponential

Syntax

Fluid Temperature [*{of}* *SpeciesName*] = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Scope

Aria Material

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Fluid Temperature [*{of} SpeciesName*] = Global (Global_Name = *global_name* | [Component = *component*])

Scope

Aria Material

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Fluid Temperature [*{of} SpeciesName*] = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Fluid Temperature [*{of}* *SpeciesName*] = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Fluid Temperature [*{of} SpeciesName*] = Scalar_String_Function F = *f*

Scope

Aria Material

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	"string"	–

User_Field

Syntax

Fluid Temperature [*{of}* *SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Fluid Temperature [*{of}* *SpeciesName*] = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Scope

Aria Material

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Fluid Temperature [*{of}* *SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	“string”	–
plugin_parameters	“string”	–

Fluid Velocity Magnitude

Correlation

Syntax

Fluid Velocity Magnitude [*{of}* *SpeciesName*] = Correlation ([V = *v*] | [Prereq_Model = *prereq_model*] | [Bulk_Node_Interface = *bulk_node_interface*] | [Epsilon = *epsilon*])

Scope

Aria Material

Summary

Correlation fluid velocity

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
v	real	0
prereq_model	“string”	ADVECTION_VELOCITY
bulk_node_interface	“string”	–
epsilon	real	1e-06

Flux Vector

Summed

Syntax

Flux Vector [*{of}* *SpeciesName*] = Summed Contributions = *contributions*

Scope

Aria Material

Summary

Flux vector as a sum of user-specified flux vectors

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
contributions	“string”	–

Forchheimer Drag Coeff

Constant

Syntax

Forchheimer Drag Coeff [*{of}* *SpeciesName*] = Constant *{c_f \| value}* = *c_f*

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
c_f	real	–

Copied

Syntax

Forchheimer Drag Coeff [*{of}* *SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	“string”	–

Exponential

Syntax

Forchheimer Drag Coeff [*{of}* *SpeciesName*] = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Scope

Aria Material

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Forchheimer Drag Coeff [*{of} SpeciesName*] = Global (Global_Name = *global_name* | [Component = *component*])

Scope

Aria Material

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Forchheimer Drag Coeff [*{of} SpeciesName*] = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Forchheimer Drag Coeff [*{of}* *SpeciesName*] = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Forchheimer Drag Coeff [*{of} SpeciesName*] = Scalar_String_Function F = *f*

Scope

Aria Material

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	"string"	–

User_Field

Syntax

Forchheimer Drag Coeff [*{of}* *SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Forchheimer Drag Coeff [*{of}* *SpeciesName*] = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Scope

Aria Material

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Forchheimer Drag Coeff [*{of} SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	“string”	–
plugin_parameters	“string”	–

Freestream Density

Constant

Syntax

Freestream Density [*{of} SpeciesName*] = Constant *{fsdens | value}* = *fsdens*

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
fsdens	real	–

Copied

Syntax

Freestream Density [*{of}* *SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	“string”	–

Exponential

Syntax

Freestream Density [*{of}* *SpeciesName*] = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Scope

Aria Material

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Freestream Density [*{of}* *SpeciesName*] = Global (Global_Name = *global_name* | [Component = *component*])

Scope

Aria Material

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Freestream Density [*{of}* *SpeciesName*] = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Freestream Density [*{of}* *SpeciesName*] = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Freestream Density [*{of} SpeciesName*] = Scalar_String_Function F = *f*

Scope

Aria Material

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	"string"	–

User_Field

Syntax

Freestream Density [*{of}* *SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Freestream Density [*{of}* *SpeciesName*] = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Scope

Aria Material

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Freestream Density [*{of} SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	“string”	–
plugin_parameters	“string”	–

Freestream Gamma

Constant

Syntax

Freestream Gamma [*{of} SpeciesName*] = Constant *{g | value}* = *g*

Scope

Aria Material

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
g	real	–

Freestream Pressure

Constant

Syntax

Freestream Pressure [*{of} SpeciesName*] = Constant *{fspres | value}* = *fspres*

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
fspres	real	–

Copied

Syntax

Freestream Pressure [*{of} SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	“string”	–

Exponential

Syntax

Freestream Pressure [*{of} SpeciesName*] = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Scope

Aria Material

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Freestream Pressure [*{of} SpeciesName*] = Global (Global_Name = *global_name* | [Component = *component*])

Scope

Aria Material

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Freestream Pressure [*{of} SpeciesName*] = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Freestream Pressure [*{of} SpeciesName*] = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Freestream Pressure [*{of} SpeciesName*] = Scalar_String_Function F = *f*

Scope

Aria Material

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	"string"	–

User_Field

Syntax

Freestream Pressure [*{of}* *SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Freestream Pressure [*{of}* *SpeciesName*] = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Scope

Aria Material

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Freestream Pressure [*{of} SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	“string”	–
plugin_parameters	“string”	–

Freestream Specific Heat

Constant

Syntax

Freestream Specific Heat [*{of} SpeciesName*] = Constant Value = *value*

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
value	real	–

Copied

Syntax

Freestream Specific Heat [*{of}* *SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	“string”	–

Exponential

Syntax

Freestream Specific Heat [*{of}* *SpeciesName*] = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Scope

Aria Material

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Freestream Specific Heat [*{of} SpeciesName*] = Global (Global_Name = *global_name* | [Component = *component*])

Scope

Aria Material

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Freestream Specific Heat [*{of} SpeciesName*] = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Freestream Specific Heat [*{of}* *SpeciesName*] = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Freestream Specific Heat [*{of} SpeciesName*] = Scalar_String_Function F = *f*

Scope

Aria Material

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	"string"	–

User_Field

Syntax

Freestream Specific Heat [*{of} SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Freestream Specific Heat [*{of} SpeciesName*] = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Scope

Aria Material

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Freestream Specific Heat [*{of} SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	“string”	–
plugin_parameters	“string”	–

Freestream Temperature

Constant

Syntax

Freestream Temperature [*{of} SpeciesName*] = Constant *{fstemp | value}* = *fstemp*

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
fstemp	real	–

Copied

Syntax

Freestream Temperature [*{of}* *SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	“string”	–

Exponential

Syntax

Freestream Temperature [*{of}* *SpeciesName*] = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Scope

Aria Material

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Freestream Temperature [*{of}* *SpeciesName*] = Global (Global_Name = *global_name* | [Component = *component*])

Scope

Aria Material

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Freestream Temperature [*{of}* *SpeciesName*] = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Freestream Temperature [*{of} SpeciesName*] = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Freestream Temperature [*{of}* *SpeciesName*] = Scalar_String_Function F = *f*

Scope

Aria Material

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	"string"	–

User_Field

Syntax

Freestream Temperature [*{of}* *SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Freestream Temperature [*{of}* *SpeciesName*] = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Scope

Aria Material

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Freestream Temperature [*{of}* *SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	“string”	–
plugin_parameters	“string”	–

Freestream Thermal Conductivity

Constant

Syntax

Freestream Thermal Conductivity [*{of}* *SpeciesName*] = Constant Value = *value*

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
value	real	–

Copied

Syntax

Freestream Thermal Conductivity [*{of}* *SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	“string”	–

Exponential

Syntax

Freestream Thermal Conductivity [*{of}* *SpeciesName*] = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Scope

Aria Material

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Freestream Thermal Conductivity [*{of}* *SpeciesName*] = Global (Global_Name = *global_name* | [Component = *component*])

Scope

Aria Material

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Freestream Thermal Conductivity [*{of}* *SpeciesName*] = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Freestream Thermal Conductivity [*{of}* *SpeciesName*] = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Freestream Thermal Conductivity [*{of} SpeciesName*] = Scalar_String_Function F = *f*

Scope

Aria Material

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	"string"	–

User_Field

Syntax

Freestream Thermal Conductivity [*{of} SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Freestream Thermal Conductivity [*{of} SpeciesName*] = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Scope

Aria Material

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Freestream Thermal Conductivity [*{of} SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	“string”	–
plugin_parameters	“string”	–

Freestream Viscosity

Constant

Syntax

Freestream Viscosity [*{of} SpeciesName*] = Constant Value = *value*

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
value	real	–

Copied

Syntax

Freestream Viscosity [*{of}* *SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	“string”	–

Exponential

Syntax

Freestream Viscosity [*{of}* *SpeciesName*] = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Scope

Aria Material

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Freestream Viscosity [*{of}* *SpeciesName*] = Global (Global_Name = *global_name* | [Component = *component*])

Scope

Aria Material

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Freestream Viscosity [*{of}* *SpeciesName*] = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Freestream Viscosity [*{of}* *SpeciesName*] = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Freestream Viscosity [*{of}* *SpeciesName*] = Scalar_String_Function F = *f*

Scope

Aria Material

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	"string"	–

User_Field

Syntax

Freestream Viscosity [*{of}* *SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Freestream Viscosity [*{of}* *SpeciesName*] = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Scope

Aria Material

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Freestream Viscosity [*{of} SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	“string”	–
plugin_parameters	“string”	–

Friction Factor

Constant

Syntax

Friction Factor [*{of} SpeciesName*] = Constant *{f | value}* = *f*

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	real	–

Copied

Syntax

Friction Factor [*{of}* *SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	“string”	–

Correlation

Syntax

Friction Factor [*{of}* *SpeciesName*] = Correlation (F = *f* | [Bulk_Node_Interface = *bulk_node_interface*])

Scope

Aria Material

Summary

Correlation friction factor with constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	real	–
bulk_node_interface	“string”	–

Correlation_Smooth_Annulus

Syntax

Friction Factor [*{of}* *SpeciesName*] = Correlation_Smooth_Annulus
[Bulk_Node_Interface = *bulk_node_interface*]

Scope

Aria Material

Summary

Correlation friction factor with smooth annulus model

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
bulk_node_interface	"string"	–

Correlation_Smooth_Tube

Syntax

Friction Factor [*{of}* *SpeciesName*] = Correlation_Smooth_Tube
[Bulk_Node_Interface = *bulk_node_interface*]

Scope

Aria Material

Summary

Correlation friction factor with smooth tube model

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
bulk_node_interface	"string"	–

Exponential

Syntax

Friction Factor [*{of}* *SpeciesName*] = Exponential (Variable = *variable* |
[Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Scope

Aria Material

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Friction Factor [*{of}* *SpeciesName*] = Global (Global_Name = *global_name* | [Component = *component*])

Scope

Aria Material

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
global_name	"string"	–
component	integer	0

Haaland

Syntax

Friction Factor [*{of}* *SpeciesName*] = Haaland Roughness = *roughness*

Scope

Aria Material

Summary

Friction factor computed using the explicit Haaland relation.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
roughness	real	–

Nd_Table

Syntax

Friction Factor [*{of} SpeciesName*] = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Friction Factor [*{of} SpeciesName*] = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Friction Factor [*{of}* *SpeciesName*] = Scalar_String_Function F = *f*

Scope

Aria Material

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	"string"	–

Smooth_Annulus

Syntax

Friction Factor [*{of} SpeciesName*] = Smooth_Annulus R = *r*

Scope

Aria Material

Summary

Friction factor computed from smooth annulus model

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
r	"string"	–

Smooth_Tube

Syntax

Friction Factor [*{of} SpeciesName*] = Smooth_Tube ()

Scope

Aria Material

Summary

Friction factor computed from smooth tube model

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–

User_Field

Syntax

Friction Factor [*{of} SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Friction Factor [*{of}* *SpeciesName*] = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Scope

Aria Material

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1
toggle	"string"	–

User_Plugin

Syntax

Friction Factor [*{of} SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	"string"	–
plugin_parameters	"string"	–

G

Gamma

Constant

Syntax

Gamma [*{of} SpeciesName*] = Constant *{g | value}* = *g*

Scope

Aria Material

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
g	real	–

Gamma Dot

Constant

Syntax

Gamma Dot [*{of}* *SpeciesName*] = Constant Value = *value*

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
value	real	–

Copied

Syntax

Gamma Dot [*{of}* *SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	“string”	–

Exponential

Syntax

Gamma Dot [*{of}* *SpeciesName*] = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Scope

Aria Material

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Gamma Dot [*{of} SpeciesName*] = Global (Global_Name = *global_name* | [Component = *component*])

Scope

Aria Material

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Gamma Dot [*{of} SpeciesName*] = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Gamma Dot [*{of} SpeciesName*] = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Gamma Dot [*{of}* *SpeciesName*] = Scalar_String_Function F = *f*

Scope

Aria Material

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	"string"	–

User_Field

Syntax

Gamma Dot [*{of}* *SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Gamma Dot [*{of}* *SpeciesName*] = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Scope

Aria Material

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Gamma Dot [*{of} SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	“string”	–
plugin_parameters	“string”	–

Gap Conductance Coefficient

Calore_User_Sub

Syntax

Gap Conductance Coefficient ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) = Calore_User_Sub (Name = *name* | Type = *type* | [Multiplier = *multiplier*] | [Material_Data_Block = *material_data_block*] | [Data = *data*] | [Scaling_Field = *scaling_field*] | [Equations = *Equations. . .*])

Scope

Aria Material

Summary

Calore user subroutine contact gap conductance

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
name	"string"	–
type	"string"	–
multiplier	real	1
material_data_block	"string"	–
data	"string"	–
scaling_field	"string"	–
Equations	string. . .	–

Constant

Syntax

Gap Conductance Coefficient ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) = Constant (*{h | value}* = *h* | [Equations = *Equations. . .*])

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
h	real	–
Equations	string. . .	–

Copied

Syntax

Gap Conductance Coefficient ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) = Copied (Source = *source* | [Equations = *Equations. . .*])

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
source	"string"	–
Equations	string. . .	–

Exponential

Syntax

Gap Conductance Coefficient ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent* | [Equations = *Equations. . .*])

Scope

Aria Material

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–
Equations	string. . .	–

Fortran

Syntax

Gap Conductance Coefficient ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) = Fortran ([Multiplier = *multiplier*] | Sub_Name = *sub_name* | [Real_Data = *real_data*] | [Int_Data = *int_data*] | [Resource_Name = *resource_name*] | [Data = *data*] | [Equations = *Equations. . .*])

Scope

Aria Material

Summary

Fortran contact gap conductance subroutine.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
multiplier	real	1
sub_name	"string"	–
real_data	"string"	–
int_data	"string"	–
resource_name	"string"	–
data	"string"	–
Equations	string. . .	–

Global

Syntax

Gap Conductance Coefficient ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) = Global (Global_Name = *global_name* | [Component = *component*] | [Equations = *Equations. . .*])

Scope

Aria Material

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
global_name	"string"	–
component	integer	0
Equations	string. . .	–

Nd_Table

Syntax

Gap Conductance Coefficient ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*] | [Equations = *Equations. . .*])

Scope

Aria Material

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–
Equations	string. . .	–

Polynomial

Syntax

Gap Conductance Coefficient ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*] | [Equations = *Equations. . .*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0
Equations	string. . .	–

Scalar_String_Function

Syntax

Gap Conductance Coefficient ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) = Scalar_String_Function (F = *f* | [Equations = *Equations. . .*])

Scope

Aria Material

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
f	"string"	–
Equations	string. . .	–

User_Field

Syntax

Gap Conductance Coefficient ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*] | [Equations = *Equations. . .*])

Scope

Aria Material

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–
Equations	string. . .	–

User_Function

Syntax

Gap Conductance Coefficient ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*] | [Equations = *Equations. . .*])

Scope

Aria Material

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1
toggle	"string"	–
Equations	string. . .	–

User_Plugin

Syntax

Gap Conductance Coefficient ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) = User_Plugin (Name = *Name* | [*plugin_parameters*] | [Equations = *Equations. . .*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Name	“string”	–
plugin_parameters	“string”	–
Equations	string. . .	–

Gap Height

Constant

Syntax

Gap Height [*{of}* *SpeciesName*] = Constant *{h | value}* = *h*

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
h	real	–

Copied

Syntax

Gap Height [*{of}* *SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	“string”	–

Exponential

Syntax

Gap Height [*{of}* *SpeciesName*] = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Scope

Aria Material

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Gap Height [*{of}* *SpeciesName*] = Global (Global_Name = *global_name* | [Component = *component*])

Scope

Aria Material

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Gap Height [*{of} SpeciesName*] = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Gap Height [*{of} SpeciesName*] = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Gap Height [*{of} SpeciesName*] = Scalar_String_Function F = *f*

Scope

Aria Material

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	"string"	–

User_Field

Syntax

Gap Height [*{of}* *SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Gap Height [*{of}* *SpeciesName*] = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Scope

Aria Material

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Gap Height [*{of} SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	“string”	–
plugin_parameters	“string”	–

Gas Phase Retardation

Constant

Syntax

Gas Phase Retardation [*{of} SpeciesName*] = Constant *{r | value}* = *r*

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
r	real	–

Copied

Syntax

Gas Phase Retardation [*{of}* *SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	“string”	–

Exponential

Syntax

Gas Phase Retardation [*{of}* *SpeciesName*] = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Scope

Aria Material

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Gas Phase Retardation [*{of}* *SpeciesName*] = Global (Global_Name = *global_name* | [Component = *component*])

Scope

Aria Material

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Gas Phase Retardation [*{of}* *SpeciesName*] = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Gas Phase Retardation [*{of}* *SpeciesName*] = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Gas Phase Retardation [*{of}* *SpeciesName*] = Scalar_String_Function F = *f*

Scope

Aria Material

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	"string"	–

User_Field

Syntax

Gas Phase Retardation [*{of}* *SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Gas Phase Retardation [*{of}* *SpeciesName*] = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Scope

Aria Material

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Gas Phase Retardation [*{of} SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	“string”	–
plugin_parameters	“string”	–

Glass Transition Temperature

Constant

Syntax

Glass Transition Temperature [*{of} SpeciesName*] = Constant Value = *value*

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
value	real	–

Copied

Syntax

Glass Transition Temperature [*{of}* *SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	“string”	–

Debenedeto

Syntax

Glass Transition Temperature [*{of}* *SpeciesName*] = Debenedeto (Extent_Species_Name = *extent_species_name* | A = *a* | Tg0 = *Tg0* | Tginfinity = *TgInfinity*)

Scope

Aria Material

Summary

deBenedeto model for the glass transition temperature based on the extent of polymerization, x, and material parameters $A, T_{g0}, T_{g\infty}$ $T_g = \frac{T_{g0}(1-x)+AxT_{g\infty}}{1-x+Ax}$.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
extent_species_name	“string”	–
a	real	–
Tg0	real	–
TgInfinity	real	–

Exponential

Syntax

Glass Transition Temperature [*{of}* *SpeciesName*] = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Scope

Aria Material

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Glass Transition Temperature [*{of}* *SpeciesName*] = Global (Global_Name = *global_name* | [Component = *component*])

Scope

Aria Material

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Glass Transition Temperature [*{of}* *SpeciesName*] = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Glass Transition Temperature [*{of}* *SpeciesName*] = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Glass Transition Temperature [*{of}* *SpeciesName*] = Scalar_String_Function F = *f*

Scope

Aria Material

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	"string"	–

User_Field

Syntax

Glass Transition Temperature [*{of}* *SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Glass Transition Temperature [*{of}* *SpeciesName*] = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Scope

Aria Material

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Glass Transition Temperature [*{of} SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	“string”	–
plugin_parameters	“string”	–

Gnielinski Film Gradient Correction

Correlation

Syntax

Gnielinski Film Gradient Correction [*{of} SpeciesName*] = Correlation (Phase = *phase* | [Gas_Exponent = *gas_exponent*] | [Bulk_Node_Interface = *bulk_node_interface*])

Scope

Aria Material

Summary

Correction factor for difference between near-wall and bulk fluid state for Gnielinski pipe and annulus correlations.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
phase	"string"	–
gas_exponent	real	0.45
bulk_node_interface	"string"	FALSE

Gravitational Constant

Correlation

Syntax

Gravitational Constant [*{of}* *SpeciesName*] = Correlation ([G = *g*] | [Bulk_Node_Interface = *bulk_node_interface*])

Scope

Aria Material

Summary

Correlation Gravitational Constant

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
g	real	0
bulk_node_interface	"string"	–

H

Hausen Entrance Effect Correction

Correlation

Syntax

Hausen Entrance Effect Correction [*{of}* *SpeciesName*] = Correlation (Max_Correction = *max_correction* | [Bulk_Node_Interface = *bulk_node_interface*])

Scope

Aria Material

Summary

Correction factor for entrance effects for Gnielinski pipe and annulus correlations.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
max_correction	real	–
bulk_node_interface	“string”	FALSE

Hdiff Flux

Constant_Cl

Syntax

Hdiff Flux [*{of}* *SpeciesName*] = Constant_Cl ([Jx = *jx*] | [Jy = *jy*] | [Jz = *jz*])

Scope

Aria Material

Summary

This model has been deprecated and removed from the code.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
jx	real	0
jy	real	0
jz	real	0

Heat Conduction

Basic

Syntax

Heat Conduction [*{of}* *SpeciesName*] = Basic ()

Scope

Aria Material

Summary

Equivalent to GENERALIZED, present for backwards compatibility.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–

Constant

Syntax

Heat Conduction [*{of}* *SpeciesName*] = Constant ([Valuex = *valuex*] | [Valuey = *valuey*] | [Valuez = *valuez*])

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
valuex	real	0
valuey	real	0
valuez	real	0

Convected_Enthalpy

Syntax

Heat Conduction [*{of}* *SpeciesName*] = Convected_Enthalpy ()

Scope

Aria Material

Summary

Convected enthalpy model for heat conduction

Description

Heat conduction is calculated as

$$\mathbf{q} = \rho h \mathbf{u}$$

where ρ is the density, h is specific enthalpy, and $\mathbf{u}$ is the velocity

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–

Copied

Syntax

Heat Conduction [*{of} SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	“string”	–

Fouriers_Law

Syntax

Heat Conduction [*{of} SpeciesName*] = Fouriers_Law ()

Scope

Aria Material

Summary

Equivalent to GENERALIZED, present for backwards compatibility.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–

Generalized

Syntax

Heat Conduction [*{of} SpeciesName*] = Generalized ()

Scope

Aria Material

Summary

Generalized model for heat conduction - can use either a scalar or tensor thermal conductivity.

Description

If a tensor thermal conductivity K is provided in the material model, the form of

heat flux is

$$\mathbf{q} = -K \cdot \nabla T$$

If the thermal conductivity is given as a scalar k then the heat flux is

$$\mathbf{q} = -k \nabla T$$

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–

Mass_Diffusion_Energy_Transport

Syntax

Heat Conduction [*{of}* *SpeciesName*] = Mass_Diffusion_Energy_Transport ()

Scope

Aria Material

Summary

Mass diffusion model for energy transport

Description

Heat conduction is calculated as

$$\mathbf{q} = \rho_g D (h_{water(g)} \nabla Y_{water(g)} + h_{air(g)} \nabla Y_{air(g)})$$

where ρ_g is the gas phase density, Y is the mass fraction, D is the mass diffusivity and h is the specific enthalpy. The subscripts $air(g)$ and $water(g)$ represent air and water in the gas phase.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–

Phase_Average

Syntax

Heat Conduction [*{of}* *SpeciesName*] = Phase_Average ()

Scope

Aria Material

Summary

Phase average model for heat conduction

Description

Heat conduction is calculated as

$$\mathbf{q} = (1 - H(\phi))\mathbf{q}_A + H(\phi)\mathbf{q}_B$$

where $H(\phi)$ is the Heaviside function based on the level set variable ϕ, $\mathbf{q}_A$ is the heat conduction in phase A, and $\mathbf{q}_B$ is the heat conduction in phase B

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–

Porous_Diffusive_Enthalpy_Flux

Syntax

Heat Conduction [*{of}* *SpeciesName*] = Porous_Diffusive_Enthalpy_Flux ()

Scope

Aria Material

Summary

Porous diffusive enthalpy flux model. This is used to calculate the heat flux through a porous material with species.

Description

Heat conduction is calculated as

$$\mathbf{q} = \phi k \nabla T + \phi \rho \sum_{i=1}^{N} D_i h_i \nabla Y_i$$

where ρ is the density, ϕ is the solid phase porosity, Y_i is the mass fraction of species i, D_i is the mass diffusivity of species i, h_i is the specific enthalpy for of species i, k is the heat conduction coefficient, and T is the temperature. The first expression is summed over N species defined in the material.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–

Porous_Simplified_Diffusive_Enthalpy_Flux

Syntax

Heat Conduction [*{of} SpeciesName*] =
Porous_Simplified_Diffusive_Enthalpy_Flux ()

Scope

Aria Material

Summary

Porous simplified diffusive enthalpy flux model

Description

Heat conduction is calculated as

$$\mathbf{q} = \phi \rho D \nabla h$$

where ρ is the density, ϕ is the solid phase porosity, D is the mixture mass diffusivity, and h is the specific enthalpy.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–

Radiative_Flux

Syntax

Heat Conduction [*{of} SpeciesName*] = Radiative_Flux ([Multiplier = *multiplier*] | Enclosure = *enclosure*)

Scope

Aria Material

Summary

Transforms the radiative heat flux calculated by Chaparral into a nodal surface heat conduction flux quantity. This is used in enclosures (*Enclosure Radiation*)

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
multiplier	real	1
enclosure	“string”	–

Scalar_Vector_Product

Syntax

Heat Conduction [*{of} SpeciesName*] = Scalar_Vector_Product (Vector = *vector* | [Scalars = *scalars*] | [Multiplier = *multiplier*])

Scope

Aria Material

Summary

Scalar vector product

Description

Multiplies a vector by a list of provided scalars (variables) and a multiplier (constant)

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
vector	"string"	–
scalars	"string"	–
multiplier	real	1

Tensor_Vector_Product

Syntax

Heat Conduction [*{of} SpeciesName*] = Tensor_Vector_Product (Vector = *vector* | Tensor = *tensor* | [Scale = *scale*])

Scope

Aria Material

Summary

Tensor vector product

Description

Calculates a vector resulting from a tensor-vector product with an optional scaling factor.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
vector	"string"	–
tensor	"string"	–
scale	real	1

Thermoelectric

Syntax

Heat Conduction [*{of} SpeciesName*] = Thermoelectric ()

Scope

Aria Material

Summary

Thermoelectric model for heat conduction

Description

Heat conduction is calculated as

$$\mathbf{q} = -k\nabla T + SJT$$

where k is the heat conduction, T is temperature, S is the Seeback coefficient, and J is the current density. The Seeback coefficient must be defined in the material.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–

User_Field

Syntax

Heat Conduction [*{of} SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Vector from User Field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
scaling	real	1
global_var	“string”	–
time_function	“string”	–

User_Function

Syntax

Heat Conduction [*{of} SpeciesName*] = User_Function ([Name_X = *name_x*] | [Name_Y = *name_y*] | [Name_Z = *name_z*] | X = *x* | [Multiplier = *multiplier*])

Scope

Aria Material

Summary

Value from a Vector User Function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name_x	"string"	–
name_y	"string"	–
name_z	"string"	–
x	"string"	–
multiplier	real	1

User_Plugin

Syntax

Heat Conduction [*{of} SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	"string"	–
plugin_parameters	"string"	–

Vector_String_Function

Syntax

Heat Conduction [*{of} SpeciesName*] = Vector_String_Function (F_X = *f_x* | F_Y = *f_y* | [F_Z = *f_z*])

Scope

Aria Material

Summary

Value from a string function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f_x	"string"	–
f_y	"string"	–
f_z	"string"	–

Heat Of Vaporization

Constant

Syntax

Heat Of Vaporization [*{of} SpeciesName*] = Constant *{hv | value}* = *hv*

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
hv	real	–

Copied

Syntax

Heat Of Vaporization [*{of}* *SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	"string"	–

Exponential

Syntax

Heat Of Vaporization [*{of}* *SpeciesName*] = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Scope

Aria Material

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Heat Of Vaporization [*{of}* *SpeciesName*] = Global (Global_Name = *global_name* | [Component = *component*])

Scope

Aria Material

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
global_name	“string”	–
component	integer	0

Nd_Table

Syntax

Heat Of Vaporization [*{of}* *SpeciesName*] = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Heat Of Vaporization [*{of}* *SpeciesName*] = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Heat Of Vaporization [*{of}* *SpeciesName*] = Scalar_String_Function F = *f*

Scope

Aria Material

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be

a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	"string"	–

User_Field

Syntax

Heat Of Vaporization [*{of}* *SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Heat Of Vaporization [*{of}* *SpeciesName*] = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Scope

Aria Material

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1
toggle	"string"	–

User_Plugin

Syntax

Heat Of Vaporization [*{of} SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	"string"	–
plugin_parameters	"string"	–

Heat Transfer Coefficient

Aero

Syntax

Heat Transfer Coefficient [*{of} SpeciesName*] = Aero (T_On = *t_on* | T_Off = *t_off* | Offset_Direction = *offset_direction* | Coord_Offset = *coord_offset* | Cone_Length = *cone_length* | Gas_Constant = *gas_constant*)

Scope

Aria Material

Summary

Heat transfer coefficient for external flow about conical tipped body accounting for flow conditions including Mach number, altitude, fluid properties and local flow regime.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
t_on	real	–
t_off	real	–
offset_direction	"string"	–
coord_offset	"string"	–
cone_length	real	–
gas_constant	real	–

Calore_User_Sub

Syntax

Heat Transfer Coefficient [*{of} SpeciesName*] = Calore_User_Sub (Name = *name* | Type = *type* | [Multiplier = *multiplier*] | [Material_Data_Block = *material_data_block*] | [Data = *data*] | [Scaling_Field = *scaling_field*])

Scope

Aria Material

Summary

Values from a Calore user subroutine

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
type	"string"	–
multiplier	real	1
material_data_block	"string"	–
data	"string"	–
scaling_field	"string"	–

Constant

Syntax

Heat Transfer Coefficient [*{of} SpeciesName*] = Constant *{h | value}* = *h*

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
h	real	–

Copied

Syntax

Heat Transfer Coefficient [*{of} SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	“string”	–

Correlation

Syntax

Heat Transfer Coefficient [*{of} SpeciesName*] = Correlation (Rex = *Rex* | [Bulk_Node_Interface = *bulk_node_interface*])

Scope

Aria Material

Summary

Correlation based heat transfer coefficient

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Rex	real	–
bulk_node_interface	"string"	–

Exponential

Syntax

Heat Transfer Coefficient [*{of}* *SpeciesName*] = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Scope

Aria Material

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Fortran

Syntax

Heat Transfer Coefficient [*{of}* *SpeciesName*] = Fortran ([Multiplier = *multiplier*] | Sub_Name = *sub_name* | [Real_Data = *real_data*] | [Int_Data = *int_data*] | [Resource_Name = *resource_name*] | [Data = *data*])

Scope

Aria Material

Summary

Fortran subroutine heat transfer coefficient.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
multiplier	real	1
sub_name	“string”	–
real_data	“string”	–
int_data	“string”	–
resource_name	“string”	–
data	“string”	–

Global

Syntax

Heat Transfer Coefficient [*{of} SpeciesName*] = Global (Global_Name = *global_name* | [Component = *component*])

Scope

Aria Material

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
global_name	“string”	–
component	integer	0

Nd_Table

Syntax

Heat Transfer Coefficient [*{of} SpeciesName*] = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Heat Transfer Coefficient [*{of} SpeciesName*] = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Heat Transfer Coefficient [*{of} SpeciesName*] = Scalar_String_Function F = *f*

Scope

Aria Material

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	"string"	–

User_Field

Syntax

Heat Transfer Coefficient [*{of}* *SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Heat Transfer Coefficient [*{of}* *SpeciesName*] = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Scope

Aria Material

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Heat Transfer Coefficient [*{of} SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	“string”	–
plugin_parameters	“string”	–

Hydraulic Diameter

Constant

Syntax

Hydraulic Diameter [*{of} SpeciesName*] = Constant *{d | value}* = *d*

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
d	real	–

Copied

Syntax

Hydraulic Diameter [*{of}* *SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	“string”	–

Correlation

Syntax

Hydraulic Diameter [*{of}* *SpeciesName*] = Correlation ([D = *d*] | [Bulk_Node_Interface = *bulk_node_interface*])

Scope

Aria Material

Summary

Correlation hydraulic diameter based on sectional area and wetted perimeter

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
d	real	0
bulk_node_interface	“string”	–

Correlation_Spatial_User_Function

Syntax

Hydraulic Diameter [*{of}* *SpeciesName*] = Correlation_Spatial_User_Function ([Threshold = *threshold*] | [Multiplier = *multiplier*] | Name = *name* | Bar = *bar* | X = *x* | [Bulk_Node_Interface = *bulk_node_interface*])

Scope

Aria Material

Summary

Correlation spatially varying hydraulic diameter based on a coordinate dependent user function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
threshold	real	-1.798e+308
multiplier	real	1
name	“string”	–
bar	“string”	–
x	“string”	–
bulk_node_interface	“string”	–

Exponential

Syntax

Hydraulic Diameter [*{of}* *SpeciesName*] = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Scope

Aria Material

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Hydraulic Diameter [*{of}* *SpeciesName*] = Global (Global_Name = *global_name* | [Component = *component*])

Scope

Aria Material

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Hydraulic Diameter [*{of}* *SpeciesName*] = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Hydraulic Diameter [*{of}* *SpeciesName*] = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Hydraulic Diameter [*{of}* *SpeciesName*] = Scalar_String_Function F = *f*

Scope

Aria Material

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	"string"	–

Spatial_User_Function

Syntax

Hydraulic Diameter [*{of}* *SpeciesName*] = Spatial_User_Function ([Threshold = *threshold*] | [Multiplier = *multiplier*] | Name = *name* | Bar = *bar* | X = *x*)

Scope

Aria Material

Summary

Spatially varying hydraulic diameter based on a coordinate dependent user function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
threshold	real	-1.798e+308
multiplier	real	1
name	"string"	–
bar	"string"	–
x	"string"	–

User_Field

Syntax

Hydraulic Diameter [*{of}* *SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Hydraulic Diameter [*{of}* *SpeciesName*] = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Scope

Aria Material

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1
toggle	"string"	–

User_Plugin

Syntax

Hydraulic Diameter [*{of}* *SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	“string”	–
plugin_parameters	“string”	–

Hydrogen Advection Velocity

Hdiff

Syntax

Hydrogen Advection Velocity [*{of}* *SpeciesName*] = Hdiff (R = *r* | V_H = *v_h*)

Scope

Aria Material

Summary

Advection velocity term for hydrogen transport

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
r	real	–
v_h	real	–

Hydrogen Diffusion Supg Tau

Shakib

Syntax

Hydrogen Diffusion Supg Tau [*{of}* *SpeciesName*] = Shakib ([H = *h*] | [Use_Advection = *use_advection*] | [Use_Diffusion = *use_diffusion*] | [Use_Time = *use_time*] | [Multiplier = *multiplier*])

Scope

Aria Material

Summary

?

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
h	real	0
use_advection	integer	1
use_diffusion	integer	1
use_time	integer	0
multiplier	real	1

Hydrostatic Pressure

Constant

Syntax

Hydrostatic Pressure [*{of}* *SpeciesName*] = Constant Value = *value*

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
value	real	–

Copied

Syntax

Hydrostatic Pressure [*{of}* *SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	“string”	–

Exponential

Syntax

Hydrostatic Pressure [*{of}* *SpeciesName*] = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Scope

Aria Material

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Hydrostatic Pressure [*{of}* *SpeciesName*] = Global (Global_Name = *global_name* | [Component = *component*])

Scope

Aria Material

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Hydrostatic Pressure [*{of} SpeciesName*] = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Hydrostatic Pressure [*{of} SpeciesName*] = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Hydrostatic Pressure [*{of}* *SpeciesName*] = Scalar_String_Function F = *f*

Scope

Aria Material

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	"string"	–

User_Field

Syntax

Hydrostatic Pressure [*{of}* *SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Hydrostatic Pressure [*{of}* *SpeciesName*] = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Scope

Aria Material

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Hydrostatic Pressure [*{of} SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	“string”	–
plugin_parameters	“string”	–

I

Induction Time Constant

Constant

Syntax

Induction Time Constant [*{of} SpeciesName*] = Constant *{tau | value}* = *tau*

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
tau	real	–

Copied

Syntax

Induction Time Constant [*{of}* *SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	“string”	–

Exponential

Syntax

Induction Time Constant [*{of}* *SpeciesName*] = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Scope

Aria Material

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Induction Time Constant [*{of} SpeciesName*] = Global (Global_Name = *global_name* | [Component = *component*])

Scope

Aria Material

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Induction Time Constant [*{of} SpeciesName*] = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Induction Time Constant [*{of}* *SpeciesName*] = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Induction Time Constant [*{of}* *SpeciesName*] = Scalar_String_Function F = *f*

Scope

Aria Material

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	"string"	–

User_Field

Syntax

Induction Time Constant [*{of}* *SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Induction Time Constant [*{of}* *SpeciesName*] = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Scope

Aria Material

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Induction Time Constant [*{of} SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	“string”	–
plugin_parameters	“string”	–

Initial Porosity

Constant

Syntax

Initial Porosity [*{of} SpeciesName*] = Constant *{phi | value}* = *phi*

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
phi	real	–

Copied

Syntax

Initial Porosity [*{of}* *SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	“string”	–

Exponential

Syntax

Initial Porosity [*{of}* *SpeciesName*] = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Scope

Aria Material

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Initial Porosity [*{of}* *SpeciesName*] = Global (Global_Name = *global_name* | [Component = *component*])

Scope

Aria Material

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Initial Porosity [*{of}* *SpeciesName*] = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Initial Porosity [*{of}* *SpeciesName*] = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Initial Porosity [*{of} SpeciesName*] = Scalar_String_Function F = *f*

Scope

Aria Material

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	"string"	–

User_Field

Syntax

Initial Porosity [*{of}* *SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Initial Porosity [*{of}* *SpeciesName*] = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Scope

Aria Material

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Initial Porosity [*{of}* *SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	“string”	–
plugin_parameters	“string”	–

Intensity Diffusion

Sp

Syntax

Intensity Diffusion [*{of}* *SpeciesName*] = Sp (Spn_Order = *SPn_order* | [Scaled = *scaled*])

Scope

Aria Material

Summary

Spherical harmonic intensity diffusion

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
SPn_order	integer	–
scaled	“string”	false

Internal Energy

Constant

Syntax

Internal Energy [*{of}* *SpeciesName*] = Constant Value = *value*

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
value	real	–

Copied

Syntax

Internal Energy [*{of}* *SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	“string”	–

Exponential

Syntax

Internal Energy [*{of}* *SpeciesName*] = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Scope

Aria Material

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Gas_Phase

Syntax

Internal Energy [*{of}* *SpeciesName*] = Gas_Phase ()

Scope

Aria Material

Summary

Gas phase model for internal energy

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–

Global

Syntax

Internal Energy [*{of}* *SpeciesName*] = Global (Global_Name = *global_name* | [Component = *component*])

Scope

Aria Material

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Internal Energy [*{of}* *SpeciesName*] = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
vars	"string"1["string"2["string"3["string"4["string"5]]]]	–
i0	"string"	–
i1	"string"	–
i2	"string"	–
x	"string"	–
y	"string"	–
z	"string"	–

Polynomial

Syntax

Internal Energy [*{of}* *SpeciesName*] = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Internal Energy [*{of}* *SpeciesName*] = Scalar_String_Function F = *f*

Scope

Aria Material

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be

a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	"string"	–

User_Field

Syntax

Internal Energy [*{of}* *SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Internal Energy [*{of}* *SpeciesName*] = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Scope

Aria Material

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1
toggle	"string"	–

User_Plugin

Syntax

Internal Energy [*{of} SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	"string"	–
plugin_parameters	"string"	–

Interphase Area

Constant

Syntax

Interphase Area [*{of} SpeciesName*] = Constant *{sa | value}* = *sa*

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
sa	real	–

Copied

Syntax

Interphase Area [*{of}* *SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	“string”	–

Exponential

Syntax

Interphase Area [*{of}* *SpeciesName*] = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Scope

Aria Material

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Interphase Area [*{of} SpeciesName*] = Global (Global_Name = *global_name* | [Component = *component*])

Scope

Aria Material

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Interphase Area [*{of} SpeciesName*] = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Interphase Area [*{of}* *SpeciesName*] = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Interphase Area [*{of} SpeciesName*] = Scalar_String_Function F = *f*

Scope

Aria Material

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	"string"	–

User_Field

Syntax

Interphase Area [*{of}* *SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Interphase Area [*{of}* *SpeciesName*] = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Scope

Aria Material

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Interphase Area [*{of}* *SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	“string”	–
plugin_parameters	“string”	–

Interphase Friction Coefficient

Wen_Yu

Syntax

Interphase Friction Coefficient [*{of}* *SpeciesName*] = Wen_Yu *{diameter | value}* = *diameter*

Scope

Aria Material

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
diameter	real	–

Intrinsic Permeability

Carmen_Kozeny

Syntax

Intrinsic Permeability [*{of}* *SpeciesName*] = Carmen_Kozeny (Sv = *Sv* | [Min_Porosity = *min_porosity*] | [Liquid_Phase_Name = *Liquid_Phase_Name*])

Scope

Aria Material

Summary

The Carmen-Kozeny model for permeability. $\kappa = \frac{1}{2\tau^2 S_v^2}\frac{\phi^3}{(1-\phi)^2}$, where S_v is the surface-to-volume ratio. This form is from B.R. Corrochano et al. Journal of Food Engineering 150 (2015) 106-116 and reduces to the traditional form if $S_v = 6/d$ and $\tau = \sqrt{(2.5)}$.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Sv	real	–
min_porosity	real	0.001
Liquid_Phase_Name	“string”	–

Constant

Syntax

Intrinsic Permeability [*{of}* *SpeciesName*] = Constant ([Xx = *xx*] | [Xy = *xy*] | [Xz = *xz*] | [Yx = *yx*] | [Yy = *yy*] | [Yz = *yz*] | [Zx = *zx*] | [Zy = *zy*] | [Zz = *zz*] | [T11 = *t11*] | [T12 = *t12*] | [T13 = *t13*] | [T21 = *t21*] | [T22 = *t22*] | [T23 = *t23*] | [T31 = *t31*] | [T32 = *t32*] | [T33 = *t33*])

Scope

Aria Material

Summary

Constant tensor value for the intrinsic permeability

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
xx	real	0
xy	real	0
xz	real	0
yx	real	0
yy	real	0
yz	real	0
zx	real	0
zy	real	0
zz	real	0
t11	real	0
t12	real	0
t13	real	0
t21	real	0
t22	real	0
t23	real	0
t31	real	0
t32	real	0
t33	real	0

Copied

Syntax

Intrinsic Permeability [*{of}* *SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	“string”	–

Diagonal

Syntax

Intrinsic Permeability [*{of}* *SpeciesName*] = Diagonal Variable = *variable*

Scope

Aria Material

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	“string”	–

Log_Reordered_Adagio

Syntax

Intrinsic Permeability [*{of}* *SpeciesName*] = Log_Reordered_Adagio Variable = *variable*

Scope

Aria Material

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	“string”	–

Reordered_Adagio

Syntax

Intrinsic Permeability [*{of}* *SpeciesName*] = Reordered_Adagio (Variable = *variable* | [Restriction = *restriction*])

Scope

Aria Material

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	“string”	–
restriction	“string”	–

Tensor_String_Function

Syntax

Intrinsic Permeability [*{of}* *SpeciesName*] = Tensor_String_Function (F_Xx = *f_xx* | F_Yy = *f_yy* | [F_Xy = *f_xy*] | [F_Zz = *f_zz*] | [F_Xz = *f_xz*] | [F_Yz = *f_yz*] | [F_Yx = *f_yx*] | [F_Zx = *f_zx*] | [F_Zy = *f_zy*])

Scope

Aria Material

Summary

Value from a string function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f_xx	“string”	–
f_yy	“string”	–
f_xy	“string”	0.0
f_zz	“string”	–
f_xz	“string”	0.0
f_yz	“string”	0.0
f_yx	“string”	0.0
f_zx	“string”	0.0
f_zy	“string”	0.0

Tensor_String_Function_Symmetric

Syntax

Intrinsic Permeability [*{of}* *SpeciesName*] = Tensor_String_Function_Symmetric (F_Xx = *f_xx* | F_Yy = *f_yy* | [F_Xy = *f_xy*] | [F_Zz = *f_zz*] | [F_Xz = *f_xz*] | [F_Yz = *f_yz*])

Scope

Aria Material

Summary

Value from a string function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f_xx	"string"	–
f_yy	"string"	–
f_xy	"string"	0.0
f_zz	"string"	–
f_xz	"string"	0.0
f_yz	"string"	0.0

User_Field

Syntax

Intrinsic Permeability [*{of}* *SpeciesName*] = User_Field (Variable = *variable* | [Restriction = *restriction*])

Scope

Aria Material

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
restriction	"string"	–

Volume_Average

Syntax

Intrinsic Permeability [*{of}* *SpeciesName*] = Volume_Average ()

Scope

Aria Material

Summary

Average intrinsic permeability: sum (volume_fractions * permeabilities) / sum (volume_fractions)

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–

Young_And_Todd

Syntax

Intrinsic Permeability [*{of} SpeciesName*] = Young_And_Todd (Pore_Diameter = *pore_diameter* | [Liquid_Phase_Name = *Liquid_Phase_Name*])

Scope

Aria Material

Summary

The Young and Todd permeability model described in SAND2010-0254.
$\kappa = \frac{d^2}{32}\frac{\phi}{\tau^2}$

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
pore_diameter	real	–
Liquid_Phase_Name	"string"	–

Intrinsic Permeability Scaling

Constant

Syntax

Intrinsic Permeability Scaling [*{of} SpeciesName*] = Constant *{s | value}* = *s*

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
s	real	–

Copied

Syntax

Intrinsic Permeability Scaling [*{of}* *SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	"string"	–

Exponential

Syntax

Intrinsic Permeability Scaling [*{of}* *SpeciesName*] = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Scope

Aria Material

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Forchheimer

Syntax

Intrinsic Permeability Scaling [*{of}* *SpeciesName*] = Forchheimer (Inertia_Coeff = *inertia_coeff* | [Gx = *gx*] | [Gy = *gy*] | [Gz = *gz*] | Xx = *xx* | Yy = *yy* | [Zz = *zz*] | [Xy = *xy*] | [Xz = *xz*] | [Yz = *yz*] | [Yx = *yx*] | [Zx = *zx*] | [Zy = *zy*] | [Fluid_Phase = *fluid_phase*])

Scope

Aria Material

Summary

Intrinsic Permeability scaling for Forchheimer flow.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
inertia_coeff	real	–
gx	real	0
gy	real	0
gz	real	0
xx	real	–
yy	real	–
zz	real	0
xy	real	0
xz	real	0
yz	real	0
yx	real	0
zx	real	0
zy	real	0
fluid_phase	“string”	–

Global

Syntax

Intrinsic Permeability Scaling [*{of}* *SpeciesName*] = Global (Global_Name = *global_name* | [Component = *component*])

Scope

Aria Material

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
global_name	“string”	–
component	integer	0

Kozeny

Syntax

Intrinsic Permeability Scaling [*{of}* *SpeciesName*] = Kozeny ([Multiplier = *multiplier*] | [Phi_Ref = *phi_ref*])

Scope

Aria Material

Summary

Intrinsic Permeability scaling function using Kozeny’s model.

Description

Scales permeability by

$$S = M \frac{(1-\phi_{ref})^2}{\phi_{ref}^3} \frac{\phi^3}{(1-\phi)^2}$$

If no value for ϕ_{ref} is provided, the scaling is

$$S = M \frac{\phi^3}{(1-\phi)^2}$$

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
multiplier	real	1
phi_ref	real	1

Nd_Table

Syntax

Intrinsic Permeability Scaling [*{of}* *SpeciesName*] = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Intrinsic Permeability Scaling [*{of} SpeciesName*] = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Intrinsic Permeability Scaling [*{of}* *SpeciesName*] = Scalar_String_Function F = *f*

Scope

Aria Material

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	"string"	–

T_Exponent

Syntax

Intrinsic Permeability Scaling [*{of}* *SpeciesName*] = T_Exponent (K_Ref = *k_ref* | T_Ref = *t_ref* | N = *n*)

Scope

Aria Material

Summary

Intrinsic permeability scaling factor as a power of normalized temperature

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
k_ref	real	–
t_ref	real	–
n	real	–

User_Field

Syntax

Intrinsic Permeability Scaling [*{of}* *SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
scaling	real	1
global_var	“string”	–
time_function	“string”	–

User_Function

Syntax

Intrinsic Permeability Scaling [*{of}* *SpeciesName*] = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Scope

Aria Material

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1
toggle	"string"	–

User_Plugin

Syntax

Intrinsic Permeability Scaling [*{of}* *SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	“string”	–
plugin_parameters	“string”	–

Inv Neutron Velocity

Constant

Syntax

Inv Neutron Velocity [*{of}* *SpeciesName*] = Constant *{invnu | value}* = *InvNu*

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
InvNu	real	–

Copied

Syntax

Inv Neutron Velocity [*{of}* *SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	“string”	–

Exponential

Syntax

Inv Neutron Velocity [*{of}* *SpeciesName*] = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Scope

Aria Material

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Inv Neutron Velocity [*{of}* *SpeciesName*] = Global (Global_Name = *global_name* | [Component = *component*])

Scope

Aria Material

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Inv Neutron Velocity [*{of}* *SpeciesName*] = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Inv Neutron Velocity [*{of}* *SpeciesName*] = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Inv Neutron Velocity [*{of}* *SpeciesName*] = Scalar_String_Function F = *f*

Scope

Aria Material

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	"string"	–

User_Field

Syntax

Inv Neutron Velocity [*{of}* *SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Inv Neutron Velocity [*{of}* *SpeciesName*] = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Scope

Aria Material

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1
toggle	"string"	–

User_Plugin

Syntax

Inv Neutron Velocity [*{of}* *SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	"string"	–
plugin_parameters	"string"	–

Irradiation

Calore_User_Sub

Syntax

Irradiation [*{of}* *SpeciesName*] = Calore_User_Sub (Name = *name* | Type = *type* | [Multiplier = *multiplier*] | [Material_Data_Block = *material_data_block*] | [Data = *data*] | [Scaling_Field = *scaling_field*])

Scope

Aria Material

Summary

Values from a Calore user subroutine

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
type	“string”	–
multiplier	real	1
material_data_block	“string”	–
data	“string”	–
scaling_field	“string”	–

Constant

Syntax

Irradiation [*{of}* *SpeciesName*] = Constant *{i | value}* = *i*

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
i	real	–

Copied

Syntax

Irradiation [*{of}* *SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	“string”	–

Exponential

Syntax

Irradiation [*{of}* *SpeciesName*] = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Scope

Aria Material

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

From_Mpmd_Field

Syntax

Irradiation [*{of}* *SpeciesName*] = From_Mpmd_Field ()

Scope

Aria Material

Summary

Irradiation values provided from MPMD coupling for the radiation transport equation

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–

Global

Syntax

Irradiation [*{of} SpeciesName*] = Global (Global_Name = *global_name* | [Component = *component*])

Scope

Aria Material

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Irradiation [*{of} SpeciesName*] = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
vars	"string"1["string"2["string"3["string"4["string"5]]]]	–
i0	"string"	–
i1	"string"	–
i2	"string"	–
x	"string"	–
y	"string"	–
z	"string"	–

Polynomial

Syntax

Irradiation [*{of}* *SpeciesName*] = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Irradiation [*{of}* *SpeciesName*] = Scalar_String_Function F = *f*

Scope

Aria Material

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be

a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	"string"	–

User_Field

Syntax

Irradiation [*{of}* *SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Irradiation [*{of}* *SpeciesName*] = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Scope

Aria Material

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1
toggle	"string"	–

User_Plugin

Syntax

Irradiation [*{of} SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	"string"	–
plugin_parameters	"string"	–

Isobaric Compressibility

Constant

Syntax

Isobaric Compressibility [*{of} SpeciesName*] = Constant *{beta | value}* = *beta*

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
beta	real	–

Copied

Syntax

Isobaric Compressibility [*{of}* *SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	“string”	–

Exponential

Syntax

Isobaric Compressibility [*{of}* *SpeciesName*] = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Scope

Aria Material

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Isobaric Compressibility [*{of} SpeciesName*] = Global (Global_Name = *global_name* | [Component = *component*])

Scope

Aria Material

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Isobaric Compressibility [*{of} SpeciesName*] = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Isobaric Compressibility [*{of}* *SpeciesName*] = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Isobaric Compressibility [*{of}* *SpeciesName*] = Scalar_String_Function F = *f*

Scope

Aria Material

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	"string"	–

User_Field

Syntax

Isobaric Compressibility [*{of}* *SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Isobaric Compressibility [*{of}* *SpeciesName*] = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Scope

Aria Material

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Isobaric Compressibility [*{of} SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	“string”	–
plugin_parameters	“string”	–

K

K Factor

Constant

Syntax

K Factor [*{of} SpeciesName*] = Constant Value = *value*

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
value	real	–

Copied

Syntax

K Factor [*{of}* *SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	“string”	–

Exponential

Syntax

K Factor [*{of}* *SpeciesName*] = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Scope

Aria Material

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

K Factor [*{of} SpeciesName*] = Global (Global_Name = *global_name* | [Component = *component*])

Scope

Aria Material

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

K Factor [*{of} SpeciesName*] = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

K Factor [*{of}* *SpeciesName*] = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

K Factor [*{of} SpeciesName*] = Scalar_String_Function F = *f*

Scope

Aria Material

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	"string"	–

User_Field

Syntax

K Factor [*{of}* *SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

K Factor [*{of}* *SpeciesName*] = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Scope

Aria Material

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

K Factor [*{of}* *SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	“string”	–
plugin_parameters	“string”	–

Kinematic Viscosity

Correlation

Syntax

Kinematic Viscosity [*{of}* *SpeciesName*] = Correlation ([Nu = *nu*] | [Bulk_Node_Interface = *bulk_node_interface*])

Scope

Aria Material

Summary

Correlation kinematic viscosity

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
nu	real	0
bulk_node_interface	“string”	–

L

Lambda Brooks Corey

Constant

Syntax

Lambda Brooks Corey [*{of}* *SpeciesName*] = Constant *{lambda | value}* = *lambda*

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
lambda	real	–

Copied

Syntax

Lambda Brooks Corey [*{of}* *SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	“string”	–

Exponential

Syntax

Lambda Brooks Corey [*{of}* *SpeciesName*] = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Scope

Aria Material

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Lambda Brooks Corey [*{of}* *SpeciesName*] = Global (Global_Name = *global_name* | [Component = *component*])

Scope

Aria Material

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Lambda Brooks Corey [*{of}* *SpeciesName*] = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Lambda Brooks Corey [*{of}* *SpeciesName*] = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Lambda Brooks Corey [*{of}* *SpeciesName*] = Scalar_String_Function F = *f*

Scope

Aria Material

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	"string"	–

User_Field

Syntax

Lambda Brooks Corey [*{of} SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Lambda Brooks Corey [*{of} SpeciesName*] = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Scope

Aria Material

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Lambda Brooks Corey [*{of} SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	“string”	–
plugin_parameters	“string”	–

Laminar Correlation Heat Transfer Coefficient

Correlation

Syntax

Laminar Correlation Heat Transfer Coefficient [*{of} SpeciesName*] = Correlation (Number = *number* | [Rex = *Rex*] | [Bulk_Node_Interface = *bulk_node_interface*])

Scope

Aria Material

Summary

Correlation based heat transfer coefficient

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
number	integer	–
Rex	real	0
bulk_node_interface	“string”	–

Laminar String Function Heat Transfer Coefficient

Correlation

Syntax

Laminar String Function Heat Transfer Coefficient [*{of} SpeciesName*] = Correlation (String_Function = *string_function* | [Bulk_Node_Interface = *bulk_node_interface*])

Scope

Aria Material

Summary

String function that can be used in place of a numbered heat transfer correlation coefficient.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
string_function	“string”	–
bulk_node_interface	“string”	FALSE

Laser Power

Constant

Syntax

Laser Power [*{of} SpeciesName*] = Constant *{p | value}* = *p*

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
p	real	–

Copied

Syntax

Laser Power [*{of}* *SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	“string”	–

Exponential

Syntax

Laser Power [*{of}* *SpeciesName*] = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Scope

Aria Material

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Laser Power [*{of} SpeciesName*] = Global (Global_Name = *global_name* | [Component = *component*])

Scope

Aria Material

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Laser Power [*{of} SpeciesName*] = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Laser Power [*{of}* *SpeciesName*] = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Laser Power [*{of} SpeciesName*] = Scalar_String_Function F = *f*

Scope

Aria Material

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	"string"	–

User_Field

Syntax

Laser Power [*{of}* *SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Laser Power [*{of}* *SpeciesName*] = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Scope

Aria Material

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Laser Power [*{of}* *SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	“string”	–
plugin_parameters	“string”	–

Latent Heat

Constant

Syntax

Latent Heat [*{of}* *SpeciesName*] = Constant Value = *value*

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
value	real	–

Copied

Syntax

Latent Heat [*{of}* *SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	“string”	–

Exponential

Syntax

Latent Heat [*{of}* *SpeciesName*] = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Scope

Aria Material

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Latent Heat [*{of} SpeciesName*] = Global (Global_Name = *global_name* | [Component = *component*])

Scope

Aria Material

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Latent Heat [*{of} SpeciesName*] = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Latent Heat [*{of}* *SpeciesName*] = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Porous_Phase_Specific_Average

Syntax

Latent Heat [*{of}* *SpeciesName*] = Porous_Phase_Specific_Average ([Liquid_Phase_Name = *Liquid_Phase_Name*] | [Gas_Phase_Name = *Gas_Phase_Name*])

Scope

Aria Material

Summary

Latent heat for a multiphase material of two or three phases

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Liquid_Phase_Name	"string"	–
Gas_Phase_Name	"string"	–

Scalar_String_Function

Syntax

Latent Heat [*{of} SpeciesName*] = Scalar_String_Function F = *f*

Scope

Aria Material

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	"string"	–

User_Field

Syntax

Latent Heat [*{of} SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Latent Heat [*{of} SpeciesName*] = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Scope

Aria Material

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Latent Heat [*{of} SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	“string”	–
plugin_parameters	“string”	–

Level Set

Constant

Syntax

Level Set [*{of}* *SpeciesName*] = Constant Value = *value*

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
value	real	–

Copied

Syntax

Level Set [*{of}* *SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	“string”	–

Exponential

Syntax

Level Set [*{of}* *SpeciesName*] = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Scope

Aria Material

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Level Set [*{of}* *SpeciesName*] = Global (Global_Name = *global_name* | [Component = *component*])

Scope

Aria Material

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Level Set [*{of}* *SpeciesName*] = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Level Set [*{of}* *SpeciesName*] = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Level Set [*{of} SpeciesName*] = Scalar_String_Function F = *f*

Scope

Aria Material

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	"string"	–

User_Field

Syntax

Level Set [*{of}* *SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Level Set [*{of}* *SpeciesName*] = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Scope

Aria Material

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Level Set [*{of} SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	“string”	–
plugin_parameters	“string”	–

Level Set Curvature

Constant

Syntax

Level Set Curvature [*{of} SpeciesName*] = Constant *{kappa | value}* = *kappa*

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
kappa	real	–

Copied

Syntax

Level Set Curvature [*{of}* *SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	“string”	–

Exponential

Syntax

Level Set Curvature [*{of}* *SpeciesName*] = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Scope

Aria Material

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Level Set Curvature [*{of} SpeciesName*] = Global (Global_Name = *global_name* | [Component = *component*])

Scope

Aria Material

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Level Set Curvature [*{of} SpeciesName*] = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Level Set Curvature [*{of}* *SpeciesName*] = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Level Set Curvature [*{of} SpeciesName*] = Scalar_String_Function F = *f*

Scope

Aria Material

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	"string"	–

User_Field

Syntax

Level Set Curvature [*{of}* *SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Level Set Curvature [*{of}* *SpeciesName*] = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Scope

Aria Material

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Level Set Curvature [*{of} SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	“string”	–
plugin_parameters	“string”	–

Level Set Diffusive Flux

Constant

Syntax

Level Set Diffusive Flux [*{of} SpeciesName*] = Constant Coeff = *coeff*

Scope

Aria Material

Summary

This model has been deprecated and removed from the code.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
coeff	real	–

Shock_Capturing

Syntax

Level Set Diffusive Flux [*{of}* *SpeciesName*] = Shock_Capturing [Multiplier = *multiplier*]

Scope

Aria Material

Summary

This model is now deprecated

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
multiplier	real	1

Unit_Gradient

Syntax

Level Set Diffusive Flux [*{of}* *SpeciesName*] = Unit_Gradient ()

Scope

Aria Material

Summary

This model has been deprecated and removed from the code.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–

Level Set Heaviside

Bf_Interpolated

Syntax

Level Set Heaviside [*{of}* *SpeciesName*] = Bf_Interpolated
[Use_Crossing_Time = *use_crossing_time*]

Scope

Aria Material

Summary

BF interpolated level set Heaviside

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
use_crossing_time	integer	–

Constant

Syntax

Level Set Heaviside [*{of}* *SpeciesName*] = Constant Value = *value*

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
value	real	–

Copied

Syntax

Level Set Heaviside [*{of}* *SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	"string"	–

Exponential

Syntax

Level Set Heaviside [*{of}* *SpeciesName*] = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Scope

Aria Material

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Level Set Heaviside [*{of}* *SpeciesName*] = Global (Global_Name = *global_name* | [Component = *component*])

Scope

Aria Material

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
global_name	“string”	–
component	integer	0

Interpolated

Syntax

Level Set Heaviside [*{of}* *SpeciesName*] = Interpolated [Epsilon = *epsilon*]

Scope

Aria Material

Summary

Interpolated level set Heaviside

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
epsilon	real	0.1

Nd_Table

Syntax

Level Set Heaviside [*{of}* *SpeciesName*] = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Level Set Heaviside [*{of}* *SpeciesName*] = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Level Set Heaviside [*{of} SpeciesName*] = Scalar_String_Function F = *f*

Scope

Aria Material

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	"string"	–

Sharp_Analytic

Syntax

Level Set Heaviside [*{of}* *SpeciesName*] = Sharp_Analytic ()

Scope

Aria Material

Summary

Sharp analytic level set Heaviside

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–

Smooth

Syntax

Level Set Heaviside [*{of}* *SpeciesName*] = Smooth [Use_Crossing_Time = *use_crossing_time*]

Scope

Aria Material

Summary

Smooth level set Heaviside

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
use_crossing_time	integer	–

User_Field

Syntax

Level Set Heaviside [*{of}* *SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Level Set Heaviside [*{of}* *SpeciesName*] = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Scope

Aria Material

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1
toggle	"string"	–

User_Plugin

Syntax

Level Set Heaviside [*{of}* *SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	"string"	–
plugin_parameters	"string"	–

Level Set Normal Bulk Conductivity

Constant

Syntax

Level Set Normal Bulk Conductivity [*{of}* *SpeciesName*] = Constant *{k | value}* = *k*

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
k	real	–

Copied

Syntax

Level Set Normal Bulk Conductivity [*{of}* *SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	"string"	–

Exponential

Syntax

Level Set Normal Bulk Conductivity [*{of}* *SpeciesName*] = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Scope

Aria Material

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Level Set Normal Bulk Conductivity [*{of}* *SpeciesName*] = Global (Global_Name = *global_name* | [Component = *component*])

Scope

Aria Material

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
global_name	"string"	–
component	integer	0

Inverse_Volume_Average

Syntax

Level Set Normal Bulk Conductivity [*{of}* *SpeciesName*] = Inverse_Volume_Average ()

Scope

Aria Material

Summary

Bulk Thermal conductivity normal to a level set field as a volume average of the species bulk densities ($\rho = \sum_i \left(\frac{v_i}{K_{b,i}}\right)^{-1}$.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–

Nd_Table

Syntax

Level Set Normal Bulk Conductivity [*{of}* *SpeciesName*] = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Level Set Normal Bulk Conductivity [*{of} SpeciesName*] = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Level Set Normal Bulk Conductivity [*{of} SpeciesName*] =
Scalar_String_Function F = *f*

Scope

Aria Material

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	"string"	–

User_Field

Syntax

Level Set Normal Bulk Conductivity [*{of} SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Level Set Normal Bulk Conductivity [*{of} SpeciesName*] = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Scope

Aria Material

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Level Set Normal Bulk Conductivity [*{of}* *SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	“string”	–
plugin_parameters	“string”	–

Volume_Average

Syntax

Level Set Normal Bulk Conductivity [*{of}* *SpeciesName*] = Volume_Average ()

Scope

Aria Material

Summary

Bulk Thermal conductivity normal to a level set field computed from volume average of species conductivities

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–

Level Set Normal Conductivity

Constant

Syntax

Level Set Normal Conductivity [*{of} SpeciesName*] = Constant *{k | value}* = *k*

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
k	real	–

Copied

Syntax

Level Set Normal Conductivity [*{of} SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	“string”	–

Exponential

Syntax

Level Set Normal Conductivity [*{of} SpeciesName*] = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Scope

Aria Material

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Level Set Normal Conductivity [*{of} SpeciesName*] = Global (Global_Name = *global_name* | [Component = *component*])

Scope

Aria Material

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Level Set Normal Conductivity [*{of} SpeciesName*] = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Level Set Normal Conductivity [*{of} SpeciesName*] = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Level Set Normal Conductivity [*{of} SpeciesName*] = Scalar_String_Function F = *f*

Scope

Aria Material

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	"string"	–

User_Field

Syntax

Level Set Normal Conductivity [*{of} SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Level Set Normal Conductivity [*{of} SpeciesName*] = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Scope

Aria Material

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1
toggle	"string"	–

User_Plugin

Syntax

Level Set Normal Conductivity [*{of} SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	"string"	–
plugin_parameters	"string"	–

Level Set Tangential Bulk Conductivity

Constant

Syntax

Level Set Tangential Bulk Conductivity [*{of} SpeciesName*] = Constant *{k | value}* = *k*

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
k	real	–

Copied

Syntax

Level Set Tangential Bulk Conductivity [*{of}* *SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	"string"	–

Exponential

Syntax

Level Set Tangential Bulk Conductivity [*{of}* *SpeciesName*] = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Scope

Aria Material

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Level Set Tangential Bulk Conductivity [*{of}* *SpeciesName*] = Global (Global_Name = *global_name* | [Component = *component*])

Scope

Aria Material

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
global_name	"string"	–
component	integer	0

Inverse_Volume_Average

Syntax

Level Set Tangential Bulk Conductivity [*{of}* *SpeciesName*] = Inverse_Volume_Average ()

Scope

Aria Material

Summary

Bulk Thermal conductivity tangential to a level set field as a volume average of the species bulk densities ($\rho = \sum_i \left(\frac{v_i}{K_{b,i}}\right)^{-1}$.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–

Nd_Table

Syntax

Level Set Tangential Bulk Conductivity [*{of}* *SpeciesName*] = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Level Set Tangential Bulk Conductivity [*{of}* *SpeciesName*] = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Level Set Tangential Bulk Conductivity [*{of}* *SpeciesName*] = Scalar_String_Function F = *f*

Scope

Aria Material

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	"string"	–

User_Field

Syntax

Level Set Tangential Bulk Conductivity [*{of}* *SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Level Set Tangential Bulk Conductivity [*{of}* *SpeciesName*] = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Scope

Aria Material

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Level Set Tangential Bulk Conductivity [*{of}* *SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	“string”	–
plugin_parameters	“string”	–

Volume_Average

Syntax

Level Set Tangential Bulk Conductivity [*{of}* *SpeciesName*] = Volume_Average ()

Scope

Aria Material

Summary

Bulk Thermal conductivity tangential to a level set field computed from volume average of species conductivities

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–

Level Set Tangential Conductivity

Constant

Syntax

Level Set Tangential Conductivity [*{of} SpeciesName*] = Constant *{k | value}* = *k*

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
k	real	–

Copied

Syntax

Level Set Tangential Conductivity [*{of} SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	"string"	–

Exponential

Syntax

Level Set Tangential Conductivity [*{of} SpeciesName*] = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Scope

Aria Material

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Level Set Tangential Conductivity [*{of}* *SpeciesName*] = Global (Global_Name = *global_name* | [Component = *component*])

Scope

Aria Material

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Level Set Tangential Conductivity [*{of}* *SpeciesName*] = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Level Set Tangential Conductivity [*{of}* *SpeciesName*] = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Level Set Tangential Conductivity [*{of}* *SpeciesName*] =
Scalar_String_Function F = *f*

Scope

Aria Material

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	"string"	–

User_Field

Syntax

Level Set Tangential Conductivity [*{of}* *SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Level Set Tangential Conductivity [*{of}* *SpeciesName*] = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Scope

Aria Material

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1
toggle	"string"	–

User_Plugin

Syntax

Level Set Tangential Conductivity [*{of} SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	"string"	–
plugin_parameters	"string"	–

Level Set Width

Constant

Syntax

Level Set Width [*{of} SpeciesName*] = Constant *{value | width}* = *width*

Scope

Aria Material

Summary

Specification of a constant width diffuse level set

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
width	real	–

Liquid Phase Retardation

Constant

Syntax

Liquid Phase Retardation [*{of}* *SpeciesName*] = Constant *{r | value}* = *r*

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
r	real	–

Copied

Syntax

Liquid Phase Retardation [*{of}* *SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	“string”	–

Exponential

Syntax

Liquid Phase Retardation [*{of}* *SpeciesName*] = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Scope

Aria Material

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Liquid Phase Retardation [*{of}* *SpeciesName*] = Global (Global_Name = *global_name* | [Component = *component*])

Scope

Aria Material

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Liquid Phase Retardation [*{of}* *SpeciesName*] = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Liquid Phase Retardation [*{of}* *SpeciesName*] = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Liquid Phase Retardation [*{of} SpeciesName*] = Scalar_String_Function F = *f*

Scope

Aria Material

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	"string"	–

User_Field

Syntax

Liquid Phase Retardation [*{of}* *SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Liquid Phase Retardation [*{of}* *SpeciesName*] = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Scope

Aria Material

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Liquid Phase Retardation [*{of}* *SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	“string”	–
plugin_parameters	“string”	–

Liquidus Temperature

Constant

Syntax

Liquidus Temperature [*{of}* *SpeciesName*] = Constant Value = *value*

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
value	real	–

Copied

Syntax

Liquidus Temperature [*{of}* *SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	“string”	–

Exponential

Syntax

Liquidus Temperature [*{of}* *SpeciesName*] = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Scope

Aria Material

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Liquidus Temperature [*{of}* *SpeciesName*] = Global (Global_Name = *global_name* | [Component = *component*])

Scope

Aria Material

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Liquidus Temperature [*{of}* *SpeciesName*] = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Liquidus Temperature [*{of}* *SpeciesName*] = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Liquidus Temperature [*{of} SpeciesName*] = Scalar_String_Function F = *f*

Scope

Aria Material

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	"string"	–

User_Field

Syntax

Liquidus Temperature [*{of}* *SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Liquidus Temperature [*{of}* *SpeciesName*] = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Scope

Aria Material

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Liquidus Temperature [*{of}* *SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	“string”	–
plugin_parameters	“string”	–

Lithium Polarization

Electrode_Object

Syntax

Lithium Polarization [*{of}* *SpeciesName*] = Electrode_Object ()

Scope

Aria Material

Summary

Polarization of lithium in electrolyte

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–

Lubrication Height

Combined

Syntax

Lubrication Height [*{of}* *SpeciesName*] = Combined ()

Scope

Aria Material

Summary

Calculate the lubrication height by combining the upper and lower heights

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–

Constant

Syntax

Lubrication Height [*{of}* *SpeciesName*] = Constant *{h | value}* = *h*

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
h	real	–

Copied

Syntax

Lubrication Height [*{of}* *SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	"string"	–

Exponential

Syntax

Lubrication Height [*{of}* *SpeciesName*] = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Scope

Aria Material

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Lubrication Height [*{of}* *SpeciesName*] = Global (Global_Name = *global_name* | [Component = *component*])

Scope

Aria Material

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
global_name	“string”	–
component	integer	0

Nd_Table

Syntax

Lubrication Height [*{of}* *SpeciesName*] = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Lubrication Height [*{of}* *SpeciesName*] = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Lubrication Height [*{of}* *SpeciesName*] = Scalar_String_Function F = *f*

Scope

Aria Material

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	"string"	–

User_Field

Syntax

Lubrication Height [*{of}* *SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Lubrication Height [*{of}* *SpeciesName*] = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Scope

Aria Material

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Lubrication Height [*{of} SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	“string”	–
plugin_parameters	“string”	–

Lubrication Height Lower

Constant

Syntax

Lubrication Height Lower [*{of} SpeciesName*] = Constant *{h_lower | value}* = *h_lower*

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
h_lower	real	–

Copied

Syntax

Lubrication Height Lower [*{of}* *SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	“string”	–

Exponential

Syntax

Lubrication Height Lower [*{of}* *SpeciesName*] = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Scope

Aria Material

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	“string”	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Lubrication Height Lower [*{of}* *SpeciesName*] = Global (Global_Name = *global_name* | [Component = *component*])

Scope

Aria Material

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
global_name	“string”	–
component	integer	0

Nd_Table

Syntax

Lubrication Height Lower [*{of}* *SpeciesName*] = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Lubrication Height Lower [*{of}* *SpeciesName*] = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Lubrication Height Lower [*{of} SpeciesName*] = Scalar_String_Function F = *f*

Scope

Aria Material

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	"string"	–

User_Field

Syntax

Lubrication Height Lower [*{of}* *SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Lubrication Height Lower [*{of}* *SpeciesName*] = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Scope

Aria Material

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Lubrication Height Lower [*{of} SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	“string”	–
plugin_parameters	“string”	–

Lubrication Height Upper

Constant

Syntax

Lubrication Height Upper [*{of} SpeciesName*] = Constant *{h_upper | value}* = *h_upper*

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
h_upper	real	–

Copied

Syntax

Lubrication Height Upper [*{of}* *SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	“string”	–

Exponential

Syntax

Lubrication Height Upper [*{of}* *SpeciesName*] = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Scope

Aria Material

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Lubrication Height Upper [*{of}* *SpeciesName*] = Global (Global_Name = *global_name* | [Component = *component*])

Scope

Aria Material

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Lubrication Height Upper [*{of}* *SpeciesName*] = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Lubrication Height Upper [*{of}* *SpeciesName*] = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Lubrication Height Upper [*{of}* *SpeciesName*] = Scalar_String_Function F = *f*

Scope

Aria Material

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	"string"	–

User_Field

Syntax

Lubrication Height Upper [*{of}* *SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Lubrication Height Upper [*{of}* *SpeciesName*] = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Scope

Aria Material

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Lubrication Height Upper [*{of} SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	“string”	–
plugin_parameters	“string”	–

Lubrication K

Constant

Syntax

Lubrication K [*{of} SpeciesName*] = Constant *{k | value}* = *k*

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
k	real	–

Copied

Syntax

Lubrication K [*{of}* *SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	“string”	–

Exponential

Syntax

Lubrication K [*{of}* *SpeciesName*] = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Scope

Aria Material

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Lubrication K [*{of}* *SpeciesName*] = Global (Global_Name = *global_name* | [Component = *component*])

Scope

Aria Material

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Lubrication K [*{of}* *SpeciesName*] = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Lubrication K [*{of}* *SpeciesName*] = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Lubrication K [*{of}* *SpeciesName*] = Scalar_String_Function F = *f*

Scope

Aria Material

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	"string"	–

User_Field

Syntax

Lubrication K [*{of}* *SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Lubrication K [*{of}* *SpeciesName*] = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Scope

Aria Material

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Lubrication K [*{of}* *SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	“string”	–
plugin_parameters	“string”	–

Lubrication Pressure

Constant

Syntax

Lubrication Pressure [*{of}* *SpeciesName*] = Constant Value = *value*

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
value	real	–

Copied

Syntax

Lubrication Pressure [*{of}* *SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	“string”	–

Exponential

Syntax

Lubrication Pressure [*{of}* *SpeciesName*] = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Scope

Aria Material

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Lubrication Pressure [*{of} SpeciesName*] = Global (Global_Name = *global_name* | [Component = *component*])

Scope

Aria Material

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Lubrication Pressure [*{of} SpeciesName*] = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Lubrication Pressure [*{of}* *SpeciesName*] = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Lubrication Pressure [*{of} SpeciesName*] = Scalar_String_Function F = *f*

Scope

Aria Material

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	"string"	–

User_Field

Syntax

Lubrication Pressure [*{of}* *SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Lubrication Pressure [*{of}* *SpeciesName*] = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Scope

Aria Material

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Lubrication Pressure [*{of}* *SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	“string”	–
plugin_parameters	“string”	–

Lubrication Velocity Lower

Constant

Syntax

Lubrication Velocity Lower [*{of}* *SpeciesName*] = Constant ([V_Lowerx = *v_lowerx*] | [V_Lowery = *v_lowery*] | [V_Lowerz = *v_lowerz*])

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
v_lowerx	real	0
v_lowery	real	0
v_lowerz	real	0

Copied

Syntax

Lubrication Velocity Lower [*{of}* *SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	“string”	–

Scalar_Vector_Product

Syntax

Lubrication Velocity Lower [*{of}* *SpeciesName*] = Scalar_Vector_Product (Vector = *vector* | [Scalars = *scalars*] | [Multiplier = *multiplier*])

Scope

Aria Material

Summary

Scalar vector product

Description

Multiplies a vector by a list of provided scalars (variables) and a multiplier (constant)

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
vector	“string”	–
scalars	“string”	–
multiplier	real	1

Tensor_Vector_Product

Syntax

Lubrication Velocity Lower [*{of}* *SpeciesName*] = Tensor_Vector_Product (Vector = *vector* | Tensor = *tensor* | [Scale = *scale*])

Scope

Aria Material

Summary

Tensor vector product

Description

Calculates a vector resulting from a tensor-vector product with an optional scaling factor.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
vector	“string”	–
tensor	“string”	–
scale	real	1

User_Field

Syntax

Lubrication Velocity Lower [*{of}* *SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Vector from User Field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Lubrication Velocity Lower [*{of} SpeciesName*] = User_Function ([Name_X = *name_x*] | [Name_Y = *name_y*] | [Name_Z = *name_z*] | X = *x* | [Multiplier = *multiplier*])

Scope

Aria Material

Summary

Value from a Vector User Function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name_x	"string"	–
name_y	"string"	–
name_z	"string"	–
x	"string"	–
multiplier	real	1

User_Plugin

Syntax

Lubrication Velocity Lower [*{of} SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	“string”	–
plugin_parameters	“string”	–

Vector_String_Function

Syntax

Lubrication Velocity Lower [*{of}* *SpeciesName*] = Vector_String_Function (F_X = *f_x* | F_Y = *f_y* | [F_Z = *f_z*])

Scope

Aria Material

Summary

Value from a string function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f_x	“string”	–
f_y	“string”	–
f_z	“string”	–

Lubrication Velocity Upper

Constant

Syntax

Lubrication Velocity Upper [*{of}* *SpeciesName*] = Constant ([V_Upperx = *v_upperx*] | [V_Uppery = *v_uppery*] | [V_Upperz = *v_upperz*])

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
v_upperx	real	0
v_uppery	real	0
v_upperz	real	0

Copied

Syntax

Lubrication Velocity Upper [*{of}* *SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	“string”	–

Scalar_Vector_Product

Syntax

Lubrication Velocity Upper [*{of}* *SpeciesName*] = Scalar_Vector_Product (Vector = *vector* | [Scalars = *scalars*] | [Multiplier = *multiplier*])

Scope

Aria Material

Summary

Scalar vector product

Description

Multiplies a vector by a list of provided scalars (variables) and a multiplier (constant)

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
vector	"string"	–
scalars	"string"	–
multiplier	real	1

Tensor_Vector_Product

Syntax

Lubrication Velocity Upper [*{of} SpeciesName*] = Tensor_Vector_Product (Vector = *vector* | Tensor = *tensor* | [Scale = *scale*])

Scope

Aria Material

Summary

Tensor vector product

Description

Calculates a vector resulting from a tensor-vector product with an optional scaling factor.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
vector	"string"	–
tensor	"string"	–
scale	real	1

User_Field

Syntax

Lubrication Velocity Upper [*{of} SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Vector from User Field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Lubrication Velocity Upper [*{of}* *SpeciesName*] = User_Function ([Name_X = *name_x*] | [Name_Y = *name_y*] | [Name_Z = *name_z*] | X = *x* | [Multiplier = *multiplier*])

Scope

Aria Material

Summary

Value from a Vector User Function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name_x	"string"	–
name_y	"string"	–
name_z	"string"	–
x	"string"	–
multiplier	real	1

User_Plugin

Syntax

Lubrication Velocity Upper [*{of}* *SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	“string”	–
plugin_parameters	“string”	–

Vector_String_Function

Syntax

Lubrication Velocity Upper [*{of} SpeciesName*] = Vector_String_Function (F_X = *f_x* | F_Y = *f_y* | [F_Z = *f_z*])

Scope

Aria Material

Summary

Value from a string function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f_x	“string”	–
f_y	“string”	–
f_z	“string”	–

M

Mach Number

Constant

Syntax

Mach Number [*{of} SpeciesName*] = Constant *{ma | value}* = *Ma*

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Ma	real	–

Copied

Syntax

Mach Number [*{of}* *SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	“string”	–

Exponential

Syntax

Mach Number [*{of}* *SpeciesName*] = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Scope

Aria Material

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Mach Number [*{of} SpeciesName*] = Global (Global_Name = *global_name* | [Component = *component*])

Scope

Aria Material

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Mach Number [*{of} SpeciesName*] = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Mach Number [*{of}* *SpeciesName*] = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Mach Number [*{of}* *SpeciesName*] = Scalar_String_Function F = *f*

Scope

Aria Material

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	"string"	–

User_Field

Syntax

Mach Number [*{of}* *SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Mach Number [*{of}* *SpeciesName*] = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Scope

Aria Material

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Mach Number [*{of}* *SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	“string”	–
plugin_parameters	“string”	–

Magnetic Field Flux

Remanent

Syntax

Magnetic Field Flux [*{of}* *SpeciesName*] = Remanent ()

Scope

Aria Material

Summary

Remanent Field Flux

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–

Magnetic Flux Intensity

Constant

Syntax

Magnetic Flux Intensity [*{of}* *SpeciesName*] = Constant Value = *value*

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
value	real	–

Copied

Syntax

Magnetic Flux Intensity [*{of}* *SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	“string”	–

Exponential

Syntax

Magnetic Flux Intensity [*{of}* *SpeciesName*] = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Scope

Aria Material

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Magnetic Flux Intensity [*{of} SpeciesName*] = Global (Global_Name = *global_name* | [Component = *component*])

Scope

Aria Material

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Magnetic Flux Intensity [*{of} SpeciesName*] = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Magnetic Flux Intensity [*{of}* *SpeciesName*] = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Magnetic Flux Intensity [*{of}* *SpeciesName*] = Scalar_String_Function F = *f*

Scope

Aria Material

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	"string"	–

User_Field

Syntax

Magnetic Flux Intensity [*{of}* *SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Magnetic Flux Intensity [*{of}* *SpeciesName*] = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Scope

Aria Material

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Magnetic Flux Intensity [*{of}* *SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	“string”	–
plugin_parameters	“string”	–

Magnetic Flux Stress

Grad_B

Syntax

Magnetic Flux Stress [*{of}* *SpeciesName*] = Grad_B ()

Scope

Aria Material

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–

Grad_Div_B

Syntax

Magnetic Flux Stress [*{of} SpeciesName*] = Grad_Div_B ()

Scope

Aria Material

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–

Magnetic_Pressure

Syntax

Magnetic Flux Stress [*{of} SpeciesName*] = Magnetic_Pressure ()

Scope

Aria Material

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–

Magnetic Permeability

Constant

Syntax

Magnetic Permeability [*{of} SpeciesName*] = Constant *{mu | value}* = *mu*

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
mu	real	–

Copied

Syntax

Magnetic Permeability [*{of}* *SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	"string"	–

Exponential

Syntax

Magnetic Permeability [*{of}* *SpeciesName*] = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Scope

Aria Material

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Magnetic Permeability [*{of}* *SpeciesName*] = Global (Global_Name = *global_name* | [Component = *component*])

Scope

Aria Material

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
global_name	“string”	–
component	integer	0

Nd_Table

Syntax

Magnetic Permeability [*{of}* *SpeciesName*] = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Magnetic Permeability [*{of}* *SpeciesName*] = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Magnetic Permeability [*{of}* *SpeciesName*] = Scalar_String_Function F = *f*

Scope

Aria Material

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be

a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	"string"	–

User_Field

Syntax

Magnetic Permeability [*{of}* *SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Magnetic Permeability [*{of}* *SpeciesName*] = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Scope

Aria Material

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The

meaning of each option is as follows:

Toggle
If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1
toggle	"string"	–

User_Plugin

Syntax
Magnetic Permeability [*{of} SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope
Aria Material

Summary
Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	"string"	–
plugin_parameters	"string"	–

Magnetic Potential Stress

Grad_A

Syntax
Magnetic Potential Stress [*{of} SpeciesName*] = Grad_A ()

Scope
Aria Material

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–

Voltage

Syntax

Magnetic Potential Stress [*{of}* *SpeciesName*] = Voltage ()

Scope

Aria Material

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–

Magnetic Pressure

Constant

Syntax

Magnetic Pressure [*{of}* *SpeciesName*] = Constant ([Toggle = *toggle*] | Value = *value*)

Scope

Aria Material

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
toggle	“string”	–
value	real	–

Copied

Syntax

Magnetic Pressure [*{of}* *SpeciesName*] = Copied ([Toggle = *toggle*] | Source = *source*)

Scope

Aria Material

Summary

Copied value from another expression

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
toggle	“string”	–
source	“string”	–

Exponential

Syntax

Magnetic Pressure [*{of}* *SpeciesName*] = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Scope

Aria Material

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Magnetic Pressure [*{of} SpeciesName*] = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Scope

Aria Material

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
toggle	"string"	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Magnetic Pressure [*{of} SpeciesName*] = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Magnetic Pressure [*{of} SpeciesName*] = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
toggle	"string"	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Magnetic Pressure [*{of}* *SpeciesName*] = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Scope

Aria Material

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name.

Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
toggle	"string"	–
f	"string"	–

User_Field

Syntax

Magnetic Pressure [*{of} SpeciesName*] = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
toggle	"string"	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Magnetic Pressure [*{of}* *SpeciesName*] = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Scope

Aria Material

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
toggle	“string”	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Magnetic Pressure [*{of}* *SpeciesName*] = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is

active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
toggle	“string”	–
Name	“string”	–
plugin_parameters	“string”	–

Magnetic Relative Permeability

Constant

Syntax

Magnetic Relative Permeability [*{of}* *SpeciesName*] = Constant Value = *value*

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
value	real	–

Copied

Syntax

Magnetic Relative Permeability [*{of}* *SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	“string”	–

Exponential

Syntax

Magnetic Relative Permeability [*{of}* *SpeciesName*] = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Scope

Aria Material

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Magnetic Relative Permeability [*{of}* *SpeciesName*] = Global (Global_Name = *global_name* | [Component = *component*])

Scope

Aria Material

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Magnetic Relative Permeability [*{of}* *SpeciesName*] = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Magnetic Relative Permeability [*{of}* *SpeciesName*] = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Magnetic Relative Permeability [*{of}* *SpeciesName*] = Scalar_String_Function F = *f*

Scope

Aria Material

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	"string"	–

User_Field

Syntax

Magnetic Relative Permeability [*{of}* *SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Magnetic Relative Permeability [*{of}* *SpeciesName*] = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Scope

Aria Material

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Magnetic Relative Permeability [*{of} SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	“string”	–
plugin_parameters	“string”	–

Magnetization

Constant

Syntax

Magnetization [*{of} SpeciesName*] = Constant ([Mx = *mx*] | [My = *my*] | [Mz = *mz*])

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
mx	real	0
my	real	0
mz	real	0

Copied

Syntax

Magnetization [*{of}* *SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	“string”	–

Scalar_Vector_Product

Syntax

Magnetization [*{of}* *SpeciesName*] = Scalar_Vector_Product (Vector = *vector* | [Scalars = *scalars*] | [Multiplier = *multiplier*])

Scope

Aria Material

Summary

Scalar vector product

Description

Multiplies a vector by a list of provided scalars (variables) and a multiplier (constant)

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
vector	"string"	–
scalars	"string"	–
multiplier	real	1

Tensor_Vector_Product

Syntax

Magnetization [*{of}* *SpeciesName*] = Tensor_Vector_Product (Vector = *vector* | Tensor = *tensor* | [Scale = *scale*])

Scope

Aria Material

Summary

Tensor vector product

Description

Calculates a vector resulting from a tensor-vector product with an optional scaling factor.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
vector	"string"	–
tensor	"string"	–
scale	real	1

User_Field

Syntax

Magnetization [*{of}* *SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Vector from User Field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Magnetization [*{of}* *SpeciesName*] = User_Function ([Name_X = *name_x*] | [Name_Y = *name_y*] | [Name_Z = *name_z*] | X = *x* | [Multiplier = *multiplier*])

Scope

Aria Material

Summary

Value from a Vector User Function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name_x	"string"	–
name_y	"string"	–
name_z	"string"	–
x	"string"	–
multiplier	real	1

User_Plugin

Syntax

Magnetization [*{of}* *SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	"string"	–
plugin_parameters	"string"	–

Vector_String_Function

Syntax

Magnetization [*{of}* *SpeciesName*] = Vector_String_Function (F_X = *f_x* | F_Y = *f_y* | [F_Z = *f_z*])

Scope

Aria Material

Summary

Value from a string function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f_x	"string"	–
f_y	"string"	–
f_z	"string"	–

Masking

Constant

Syntax

Masking [*{of}* *SpeciesName*] = Constant Value = *value*

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
value	real	–

Copied

Syntax

Masking [*{of}* *SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	"string"	–

Exponential

Syntax

Masking [*{of}* *SpeciesName*] = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Scope

Aria Material

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Masking [*{of}* *SpeciesName*] = Global (Global_Name = *global_name* | [Component = *component*])

Scope

Aria Material

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
global_name	“string”	–
component	integer	0

Nd_Table

Syntax

Masking [*{of}* *SpeciesName*] = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Masking [*{of}* *SpeciesName*] = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Masking [*{of}* *SpeciesName*] = Scalar_String_Function F = *f*

Scope

Aria Material

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be

a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	"string"	–

User_Field

Syntax

Masking [*{of}* *SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Masking [*{of}* *SpeciesName*] = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Scope

Aria Material

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Masking [*{of} SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	“string”	–
plugin_parameters	“string”	–

Mass Balance Advective Flux

Porous

Syntax

Mass Balance Advective Flux [*{of} SpeciesName*] = Porous ()

Scope

Aria Material

Summary

Provides the mass balance advection term for a given species i and phase p

Description

For species i, phase p this model provides the advective term for the mass balance equation

$$\mathbf{m}_{i,p} = Y_{i,p}\mathbf{m}_p,$$

where $Y_{i,p}$ is the species mass fraction and $\mathbf{m}_p$ is the mass flux for the given phase.

Note that the mass flux $\mathbf{m}_p$ is automatically provided from the `enthalpy advection = porous` model if not previously provided. Alternatively, one could manually specify it by using the `mass_balance_advective_flux = single_phase` model for the given material phase. See the descriptions of either model for details regarding the phase mass flux.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–

Porous_Muscl

Syntax

Mass Balance Advective Flux [*{of}* *SpeciesName*] = Porous_Muscl ([Limiter = *limiter*] | [Zeta = *zeta*] | [Use_Peclet_Blending = *use_peclet_blending*])

Scope

Aria Material

Summary

MUSCL scheme for advective flux in porous equations. Currently only valid for EQ porous_enthalpy / mass_balance while using CVFEM.

Description

Let q be the subcontrol surface interpolated scalar and let q_{muscl} be the MUSCL scalar at the subcontrol surface computed using the projected nodal gradient of the scalar and the prescribed limiter (i.e., `NONE`, `SUPERBEE`, `MINDMOD`, `VAN_ALBADA`, `VAN_LEER`). The scalar at the subcontrol surface is linearly interpolated as

$$q_f = \chi q_{muscl} + (1 - \chi)\, q,$$

where $\chi = \dfrac{(\zeta Pe)^2}{5 + (\zeta Pe)^2}$ is the weighting function between 0 and 1 that is based on the cell Peclet number Pe and ζ, a user specified scaling. If no diffusion is present, (i.e., pure advection), q_{muscl} is used. Similarly, if the cell Peclet number is near zero, then the interpolated scalar q is used.

The cell Peclet number is the ratio of advective flux to diffusive flux in a cell. For the mass balance equation, it is assumed the `MASS_BALANCE_DIFFUSIVE_FLUX = POROUS` is used, giving a diffusive flux (if present) of the form $\mathbf{J} = -\phi\rho D_i \nabla Y_i$, which in turn yields the following cell Pe

$$Pe_{\Delta x} = \frac{\mathbf{u} \cdot \mathbf{\Delta x}}{\phi D_i}.$$

On the other hand, for the gas phase porous enthalpy equation, it is assumed the `HEAT CONDUCTION = POROUS_SIMPLIFIED_DIFFUSIVE_ENTHALPY` model is used, giving a diffusive flux (if present) of the form $\mathbf{q} = -\phi\rho D \nabla h$, which in turn for a unit Lewis number yields the following cell Pe,

$$Pe_{\Delta x} = \frac{\mathbf{u} \cdot \mathbf{\Delta x}}{\phi\alpha}$$

For the solid phase (or no phase) porous enthalpy equation, it is assumed `HEAT CONDUCTION = FOURIERS_LAW` is used i.e, $\mathbf{q} = -k\nabla T$, resulting in the following cell Pe

$$Pe_{\Delta x} = \frac{\mathbf{u} \cdot \mathbf{\Delta x}}{\alpha}.$$

Note the thermal diffusivity is computed using a scalar thermal conductivity. For other cases, set the optional parameter `use_pecleting_blending = 0`, to turn off Peclet blending.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
limiter	"string"	VAN_LEER
zeta	real	1
use_peclet_blending	integer	–

Porous_Muscl_Density

Syntax

Mass Balance Advective Flux [*{of}* *SpeciesName*] = Porous_Muscl_Density ([Limiter = *limiter*] | [Zeta = *zeta*] | [Use_Peclet_Blending = *use_peclet_blending*])

Scope

Aria Material

Summary

MUSCL scheme for advective flux in porous equations. Currently only valid for EQ porous_enthalpy / mass_balance while using CVFEM.

Description

Let q be the subcontrol surface interpolated scalar and let q_{muscl} be the MUSCL scalar at the subcontrol surface computed using the projected nodal gradient of the scalar and the prescribed limiter (i.e., `NONE`, `SUPERBEE`, `MINDMOD`, `VAN_ALBADA`, `VAN_LEER`). The scalar at the subcontrol surface is linearly interpolated as

$$q_f = \chi q_{muscl} + (1 - \chi)\, q,$$

where $\chi = \dfrac{(\zeta Pe)^2}{5 + (\zeta Pe)^2}$ is the weighting function between 0 and 1 that is based on the cell Peclet number Pe and ζ, a user specified scaling. If no diffusion is present, (i.e., pure advection), q_{muscl} is used. Similarly, if the cell Peclet number is near zero, then the interpolated scalar q is used.

The cell Peclet number is the ratio of advective flux to diffusive flux in a cell. For the mass balance equation, it is assumed the `MASS_BALANCE_DIFFUSIVE_FLUX = POROUS` is used, giving a diffusive flux (if present) of the form $\mathbf{J} = -\phi\rho D_i \nabla Y_i$, which in turn yields the following cell Pe

$$Pe_{\Delta x} = \frac{\mathbf{u} \cdot \mathbf{\Delta x}}{\phi D_i}.$$

On the other hand, for the gas phase porous enthalpy equation, it is assumed the `HEAT CONDUCTION = POROUS_SIMPLIFIED_DIFFUSIVE_ENTHALPY` model is used, giving a diffusive flux (if present) of the form $\mathbf{q} = -\phi\rho D \nabla h$, which in turn for a unit Lewis number yields the following cell Pe,

$$Pe_{\Delta x} = \frac{\mathbf{u} \cdot \mathbf{\Delta x}}{\phi \alpha}$$

For the solid phase (or no phase) porous enthalpy equation, it is assumed `HEAT CONDUCTION = FOURIERS_LAW` is used i.e, $\mathbf{q} = -k\nabla T$, resulting in the following cell Pe

$$Pe_{\Delta x} = \frac{\mathbf{u} \cdot \mathbf{\Delta x}}{\alpha}.$$

Note the thermal diffusivity is computed using a scalar thermal conductivity. For other cases, set the optional parameter `use_pecleting_blending = 0`, to turn off Peclet blending.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
limiter	"string"	VAN_LEER
zeta	real	1
use_peclet_blending	integer	–

Porous_Upwind

Syntax

Mass Balance Advective Flux [*{of}* *SpeciesName*] = Porous_Upwind ()

Scope

Aria Material

Summary

Upwind mass balance advection term using the mass fraction. Only valid for CVFEM.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–

Porous_Upwind_Density

Syntax

Mass Balance Advective Flux [*{of}* *SpeciesName*] = Porous_Upwind_Density ()

Scope

Aria Material

Summary

Upwind mass balance advection term using the density. Only valid for CVFEM.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–

Single_Phase

Syntax

Mass Balance Advective Flux [*{of}* *SpeciesName*] = Single_Phase ()

Scope

Aria Material

Summary

Provides the mass balance advection term for a given species i and phase p.

Description

For species i, phase p this model provides the advective term for the mass

balance equation. By default, the non-relative form is used

$$\mathbf{m}_{i,p} = \rho_{i,p}\mathbf{u}_p,$$

where $\rho_{i,p}$ is the species density, and $\mathbf{u}_p$ is the velocity for the given phase. The non-relative form is applicable to cases with or without mesh motion. If mesh motion is detected, the mass balance equation will automatically include an appropriate volumetric mesh motion correction term to correct the advective term.

If the `use relative mass flux` porous flow option is active and mesh motion is detected, the following relative advective form is used for phases other than the solid phase (e.g., liquid / gas)

$$\mathbf{m}_{i,p} = \rho_{i,p}\left(\mathbf{u}_p - \phi S_p \hat{\mathbf{u}}\right),$$

where ϕ is the porosity, S_p is the optional phase saturation, and $\hat{\mathbf{u}}$ is the mesh velocity. If an optional phase saturation is provided, the pore volume is assumed to be occupied by multiple phases (e.g., liquid, gas) in accordance to their saturations.

Lastly, if applied to the solid phase, the relative mass flux form drops the porous mesh scaling resulting in the following relative mass flux

$$\mathbf{m}_{i,s} = \rho_{i,p}\left(\mathbf{u}_s - \hat{\mathbf{u}}\right).$$

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–

Mass Balance Diffusive Flux

Basic

Syntax

Mass Balance Diffusive Flux [*{of}* *SpeciesName*] = Basic ()

Scope

Aria Material

Summary

Mass balance diffusive term from mass diffusivity, mass fraction and density.

Description

If a tensor mass diffusivity D_i is provided in the material model, the form of diffusive flux is

$$\mathbf{J} = -\rho D_i \cdot \nabla Y_i$$

where ϕ is the porosity.

If the mass diffusivity is given as a scalar d_i then the diffusive flux is

$$\mathbf{J} = -\rho d_i \nabla Y_i$$

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–

Density

Syntax

Mass Balance Diffusive Flux [*{of}* *SpeciesName*] = Density ()

Scope

Aria Material

Summary

Mass balance diffusive term from porosity, mass diffusivity, mass fraction and density.

Description

If a tensor mass diffusivity D_i is provided in the material model, the form of diffusive flux is

$$\mathbf{J} = -\phi D_i \cdot \nabla \rho_i$$

where ϕ is the porosity.

If the mass diffusivity is given as a scalar d_i then the diffusive flux is

$$\mathbf{J} = -\phi d_i \nabla \rho_i$$

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–

Nernst_Planck

Syntax

Mass Balance Diffusive Flux [*{of}* *SpeciesName*] = Nernst_Planck ([F = *f*] | [R = *r*])

Scope

Aria Material

Description

The diffusive flux is

$$\mathbf{J} = C\left(\frac{v_i S_i F D_i}{RT}\nabla V - D\nabla S_i\right)$$

where v_i is the valence of the ion species, S_i the molar concentration, F the provided Faraday's constant, D_i the SPECIES_DIFFUSIVITY, R the provided gas constant, T the temperature, and V the voltage.

If both POROSITY and TORTUOSITY_FACTOR models are present then $C = \phi\tau$, else $C = 1$.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	real	9.649e+04
r	real	8.314

Porous

Syntax

Mass Balance Diffusive Flux [*{of}* *SpeciesName*] = Porous ()

Scope

Aria Material

Summary

Mass balance diffusive term from porosity, mass diffusivity, mass fraction and density.

Description

If a tensor mass diffusivity D_i is provided in the material model, the form of diffusive flux is

$$\mathbf{J} = -\phi\rho D_i \cdot \nabla Y_i$$

where ϕ is the porosity.

If the mass diffusivity is given as a scalar d_i then the diffusive flux is

$$\mathbf{J} = -\phi\rho d_i \nabla Y_i$$

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–

Mass Diffusivity

Air_Water

Syntax

Mass Diffusivity [*{of}* *SpeciesName*] = Air_Water (Tortuosity = *tortuosity* | P_Ref = *p_ref* | D_Ref = *d_ref* | Exponent = *exponent*)

Scope

Aria Material

Summary

Air/Water mass diffusivity

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
tortuosity	real	–
p_ref	real	–
d_ref	real	–
exponent	real	–

Arrhenius

Syntax

Mass Diffusivity [*{of}* *SpeciesName*] = Arrhenius (R = *r* | [C = *c*] | [D = *d*] | Q = *q*)

Scope

Aria Material

Summary

Arrhenius mass diffusivity, $D_m = C + D * exp\left(-\frac{Q}{RT}\right)$

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
r	real	–
c	real	0
d	real	1
q	real	–

Cantera

Syntax

Mass Diffusivity [*{of}* *SpeciesName*] = Cantera [Poffset = *poffset*]

Scope

Aria Material

Summary

Cantera mass diffusivity

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
poffset	real	0

Chapman_Enskog

Syntax

Mass Diffusivity [*{of}* *SpeciesName*] = Chapman_Enskog (D0 = *d0* \| P0 = *p0* \| T0 = *t0*)

Scope

Aria Material

Summary

Mass diffusivity model for gases that scales with $T^{3/2}/P$ based on Chapman-Enskog theory.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
d0	real	–
p0	real	–
t0	real	–

Constant

Syntax

Mass Diffusivity [*{of} SpeciesName*] = Constant *{d | value}* = *d*

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
d	real	–

Copied

Syntax

Mass Diffusivity [*{of} SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	“string”	–

Exponential

Syntax

Mass Diffusivity [*{of} SpeciesName*] = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Scope

Aria Material

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

From_Schmidt

Syntax

Mass Diffusivity [*{of}* *SpeciesName*] = From_Schmidt ()

Scope

Aria Material

Summary

Mass diffusivity from a Schmidt number and viscosity

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–

Global

Syntax

Mass Diffusivity [*{of}* *SpeciesName*] = Global (Global_Name = *global_name* | [Component = *component*])

Scope

Aria Material

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
global_name	“string”	–
component	integer	0

Nd_Table

Syntax

Mass Diffusivity [*{of}* *SpeciesName*] = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Mass Diffusivity [*{of}* *SpeciesName*] = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Mass Diffusivity [*{of}* *SpeciesName*] = Scalar_String_Function F = *f*

Scope

Aria Material

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	"string"	–

User_Field

Syntax

Mass Diffusivity [*{of}* *SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Mass Diffusivity [*{of}* *SpeciesName*] = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Scope

Aria Material

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Mass Diffusivity [*{of}* *SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	“string”	–
plugin_parameters	“string”	–

Mass Flow

Polynomial

Syntax

Mass Flow [*{of}* *SpeciesName*] = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Scope

Aria Material

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Mass Flow Rate

Global

Syntax

Mass Flow Rate [*{of}* *SpeciesName*] = Global (Global_Name = *global_name* | [Component = *component*])

Scope

Aria Material

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
global_name	"string"	–
component	integer	0

Mass Flux

Constant

Syntax

Mass Flux [*{of}* *SpeciesName*] = Constant ([Valuex = *valuex*] | [Valuey = *valuey*] | [Valuez = *valuez*])

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
valuex	real	0
valuey	real	0
valuez	real	0

Copied

Syntax

Mass Flux [*{of}* *SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	“string”	–

Darcy

Syntax

Mass Flux [*{of}* *SpeciesName*] = Darcy ()

Scope

Aria Material

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–

Scalar_Vector_Product

Syntax

Mass Flux [*{of}* *SpeciesName*] = Scalar_Vector_Product (Vector = *vector* | [Scalars = *scalars*] | [Multiplier = *multiplier*])

Scope

Aria Material

Summary

Scalar vector product

Description

Multiplies a vector by a list of provided scalars (variables) and a multiplier (constant)

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
vector	“string”	–
scalars	“string”	–
multiplier	real	1

Tensor_Vector_Product

Syntax

Mass Flux [*{of}* *SpeciesName*] = Tensor_Vector_Product (Vector = *vector* | Tensor = *tensor* | [Scale = *scale*])

Scope

Aria Material

Summary

Tensor vector product

Description

Calculates a vector resulting from a tensor-vector product with an optional scaling factor.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
vector	"string"	–
tensor	"string"	–
scale	real	1

User_Field

Syntax

Mass Flux [*{of}* *SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Vector from User Field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Mass Flux [*{of}* *SpeciesName*] = User_Function ([Name_X = *name_x*] | [Name_Y = *name_y*] | [Name_Z = *name_z*] | X = *x* | [Multiplier = *multiplier*])

Scope

Aria Material

Summary

Value from a Vector User Function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name_x	"string"	–
name_y	"string"	–
name_z	"string"	–
x	"string"	–
multiplier	real	1

User_Plugin

Syntax

Mass Flux [*{of}* *SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	"string"	–
plugin_parameters	"string"	–

Vector_String_Function

Syntax

Mass Flux [*{of}* *SpeciesName*] = Vector_String_Function (F_X = *f_x* | F_Y = *f_y* | [F_Z = *f_z*])

Scope

Aria Material

Summary

Value from a string function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f_x	“string”	–
f_y	“string”	–
f_z	“string”	–

Mass Fraction

Constant

Syntax

Mass Fraction [*{of} SpeciesName*] = Constant Value = *value*

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
value	real	–

Copied

Syntax

Mass Fraction [*{of} SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	“string”	–

Exponential

Syntax

Mass Fraction [*{of} SpeciesName*] = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Scope

Aria Material

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Fracbal

Syntax

Mass Fraction [*{of} SpeciesName*] = Fracbal ()

Scope

Aria Material

Summary

Mass fraction of one species to yield a unity sum over all species

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–

From_Chemeq

Syntax

Mass Fraction [*{of}* *SpeciesName*] = From_Chemeq ()

Scope

Aria Material

Summary

Mass fraction of one species extracted from ChemEq

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–

From_Density

Syntax

Mass Fraction [*{of}* *SpeciesName*] = From_Density ()

Scope

Aria Material

Summary

Defines mass fraction from species density and overall density

Description

Defines the mass fraction for species i as

$$Y_i = \frac{\rho_i}{\rho}$$

where ρ is the overall `DENSITY` expression for the current phase, and ρ_i is the `DENSITY` for species i in the current phase.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–

Global

Syntax

Mass Fraction [*{of} SpeciesName*] = Global (Global_Name = *global_name* | [Component = *component*])

Scope

Aria Material

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
global_name	“string”	–
component	integer	0

Nd_Table

Syntax

Mass Fraction [*{of} SpeciesName*] = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Mass Fraction [*{of}* *SpeciesName*] = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Mass Fraction [*{of}* *SpeciesName*] = Scalar_String_Function F = *f*

Scope

Aria Material

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be

a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	"string"	–

User_Field

Syntax

Mass Fraction [*{of} SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Mass Fraction [*{of} SpeciesName*] = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Scope

Aria Material

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Mass Fraction [*{of} SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	“string”	–
plugin_parameters	“string”	–

Mass Fraction Diffusive Flux

Basic

Syntax

Mass Fraction Diffusive Flux [*{of} SpeciesName*] = Basic ()

Scope

Aria Material

Summary

Basic model of mass fraction diffusive flux

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–

Energy_Mass

Syntax

Mass Fraction Diffusive Flux [*{of}* *SpeciesName*] = Energy_Mass ()

Scope

Aria Material

Summary

Energy_Mass model of mass fraction diffusive flux

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–

Material Data Block

Resource

Syntax

Material Data Block [*{of}* *SpeciesName*] = Resource Data_Block_Name = *data_block_name*

Scope

Aria Material

Summary

Resource material data block

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
data_block_name	“string”	–

Melt Temperature

Constant

Syntax

Melt Temperature [*{of}* *SpeciesName*] = Constant Value = *value*

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
value	real	–

Copied

Syntax

Melt Temperature [*{of}* *SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	“string”	–

Exponential

Syntax

Melt Temperature [*{of}* *SpeciesName*] = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Scope

Aria Material

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

From_Solidus_Liquidus

Syntax

Melt Temperature [*{of}* *SpeciesName*] = From_Solidus_Liquidus ()

Scope

Aria Material

Summary

Melt temperature as the average of the solidus and liquidus temperatures

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–

Global

Syntax

Melt Temperature [*{of}* *SpeciesName*] = Global (Global_Name = *global_name* | [Component = *component*])

Scope

Aria Material

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
global_name	“string”	–
component	integer	0

Nd_Table

Syntax

Melt Temperature [*{of} SpeciesName*] = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Melt Temperature [*{of} SpeciesName*] = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Melt Temperature [*{of} SpeciesName*] = Scalar_String_Function F = *f*

Scope

Aria Material

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	"string"	–

User_Field

Syntax

Melt Temperature [*{of}* *SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Melt Temperature [*{of}* *SpeciesName*] = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Scope

Aria Material

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Melt Temperature [*{of}* *SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	“string”	–
plugin_parameters	“string”	–

Mesh Body Acceleration

Constant

Syntax

Mesh Body Acceleration [*{of}* *SpeciesName*] = Constant ([Valuex = *valuex*] | [Valuey = *valuey*] | [Valuez = *valuez*])

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
valuex	real	0
valuey	real	0
valuez	real	0

Copied

Syntax

Mesh Body Acceleration [*{of}* *SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	“string”	–

Scalar_Vector_Product

Syntax

Mesh Body Acceleration [*{of}* *SpeciesName*] = Scalar_Vector_Product (Vector = *vector* | [Scalars = *scalars*] | [Multiplier = *multiplier*])

Scope

Aria Material

Summary

Scalar vector product

Description

Multiplies a vector by a list of provided scalars (variables) and a multiplier (constant)

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
vector	"string"	–
scalars	"string"	–
multiplier	real	1

Tensor_Vector_Product

Syntax

Mesh Body Acceleration [*{of}* *SpeciesName*] = Tensor_Vector_Product (Vector = *vector* | Tensor = *tensor* | [Scale = *scale*])

Scope

Aria Material

Summary

Tensor vector product

Description

Calculates a vector resulting from a tensor-vector product with an optional scaling factor.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
vector	"string"	–
tensor	"string"	–
scale	real	1

User_Field

Syntax

Mesh Body Acceleration [*{of}* *SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Vector from User Field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Mesh Body Acceleration [*{of} SpeciesName*] = User_Function ([Name_X = *name_x*] | [Name_Y = *name_y*] | [Name_Z = *name_z*] | X = *x* | [Multiplier = *multiplier*])

Scope

Aria Material

Summary

Value from a Vector User Function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name_x	"string"	–
name_y	"string"	–
name_z	"string"	–
x	"string"	–
multiplier	real	1

User_Plugin

Syntax

Mesh Body Acceleration [*{of} SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	“string”	–
plugin_parameters	“string”	–

Vector_String_Function

Syntax

Mesh Body Acceleration [*{of}* *SpeciesName*] = Vector_String_Function (F_X = *f_x* | F_Y = *f_y* | [F_Z = *f_z*])

Scope

Aria Material

Summary

Value from a string function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f_x	“string”	–
f_y	“string”	–
f_z	“string”	–

Mesh Displacements

Constant

Syntax

Mesh Displacements [*{of}* *SpeciesName*] = Constant ([X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
x	real	0
y	real	0
z	real	0

Copied

Syntax

Mesh Displacements [*{of}* *SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	"string"	–

User_Field

Syntax

Mesh Displacements [*{of}* *SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Vector from User Field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Mesh Displacements [*{of}* *SpeciesName*] = User_Function ([Name_X = *name_x*] | [Name_Y = *name_y*] | [Name_Z = *name_z*] | X = *x* | [Multiplier = *multiplier*])

Scope

Aria Material

Summary

Value from a Vector User Function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name_x	"string"	–
name_y	"string"	–
name_z	"string"	–
x	"string"	–
multiplier	real	1

User_Plugin

Syntax

Mesh Displacements [*{of}* *SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	"string"	–
plugin_parameters	"string"	–

Vector_String_Function

Syntax

Mesh Displacements [*{of}* *SpeciesName*] = Vector_String_Function (F_X = *f_x* | F_Y = *f_y* | [F_Z = *f_z*])

Scope

Aria Material

Summary

Value from a string function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f_x	"string"	–
f_y	"string"	–
f_z	"string"	–

Mesh Displacements X

Constant

Syntax

Mesh Displacements X [*{of}* *SpeciesName*] = Constant Value = *value*

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
value	real	–

Copied

Syntax

Mesh Displacements X [*{of}* *SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	"string"	–

Exponential

Syntax

Mesh Displacements X [*{of}* *SpeciesName*] = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Scope

Aria Material

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Mesh Displacements X [*{of}* *SpeciesName*] = Global (Global_Name = *global_name* | [Component = *component*])

Scope

Aria Material

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
global_name	“string”	–
component	integer	0

Nd_Table

Syntax

Mesh Displacements X [*{of}* *SpeciesName*] = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Mesh Displacements X [*{of}* *SpeciesName*] = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Mesh Displacements X [*{of}* *SpeciesName*] = Scalar_String_Function F = *f*

Scope

Aria Material

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be

a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	"string"	–

User_Field

Syntax

Mesh Displacements X [*{of}* *SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Mesh Displacements X [*{of}* *SpeciesName*] = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Scope

Aria Material

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The

meaning of each option is as follows:

Toggle
If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1
toggle	"string"	–

User_Plugin

Syntax
Mesh Displacements X [*{of} SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope
Aria Material

Summary
Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	"string"	–
plugin_parameters	"string"	–

Mesh Displacements Y

Constant

Syntax
Mesh Displacements Y [*{of} SpeciesName*] = Constant Value = *value*

Scope
Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
value	real	–

Copied

Syntax

Mesh Displacements Y [*{of}* *SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	“string”	–

Exponential

Syntax

Mesh Displacements Y [*{of}* *SpeciesName*] = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Scope

Aria Material

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Mesh Displacements Y [*{of}* *SpeciesName*] = Global (Global_Name = *global_name* | [Component = *component*])

Scope

Aria Material

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Mesh Displacements Y [*{of}* *SpeciesName*] = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Mesh Displacements Y [*{of}* *SpeciesName*] = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Mesh Displacements Y [*{of}* *SpeciesName*] = Scalar_String_Function F = *f*

Scope

Aria Material

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	"string"	–

User_Field

Syntax

Mesh Displacements Y [*{of}* *SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Mesh Displacements Y [*{of}* *SpeciesName*] = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Scope

Aria Material

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Mesh Displacements Y [*{of}* *SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	“string”	–
plugin_parameters	“string”	–

Mesh Displacements Z

Constant

Syntax

Mesh Displacements Z [*{of}* *SpeciesName*] = Constant Value = *value*

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
value	real	–

Copied

Syntax

Mesh Displacements Z [*{of}* *SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	“string”	–

Exponential

Syntax

Mesh Displacements Z [*{of}* *SpeciesName*] = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Scope

Aria Material

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Mesh Displacements Z [*{of}* *SpeciesName*] = Global (Global_Name = *global_name* | [Component = *component*])

Scope

Aria Material

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Mesh Displacements Z [*{of}* *SpeciesName*] = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Mesh Displacements Z [*{of}* *SpeciesName*] = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Mesh Displacements Z [*{of}* *SpeciesName*] = Scalar_String_Function F = *f*

Scope

Aria Material

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	"string"	–

User_Field

Syntax

Mesh Displacements Z [*{of}* *SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Mesh Displacements Z [*{of}* *SpeciesName*] = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Scope

Aria Material

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Mesh Displacements Z [*{of} SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	“string”	–
plugin_parameters	“string”	–

Mesh Lambda

Constant

Syntax

Mesh Lambda [*{of} SpeciesName*] = Constant *{lambda | value}* = *lambda*

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
lambda	real	–

Converted

Syntax

Mesh Lambda [*{of}* *SpeciesName*] = Converted ()

Scope

Aria Material

Summary

Lame coefficient computed from Poisson's ratio and elastic modulus

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–

Converted_Plane_Stress

Syntax

Mesh Lambda [*{of}* *SpeciesName*] = Converted_Plane_Stress ()

Scope

Aria Material

Summary

This model has been deprecated and removed from the code.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–

Copied

Syntax

Mesh Lambda [*{of}* *SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	“string”	–

Elemental_Volume

Syntax

Mesh Lambda [*{of}* *SpeciesName*] = Elemental_Volume ()

Scope

Aria Material

Summary

Lambda computed from elemental volume

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–

Exponential

Syntax

Mesh Lambda [*{of}* *SpeciesName*] = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Scope

Aria Material

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the

constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Mesh Lambda [*{of}* *SpeciesName*] = Global (Global_Name = *global_name* | [Component = *component*])

Scope

Aria Material

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
global_name	"string"	–
component	integer	0

Inverse_Element_Volume

Syntax

Mesh Lambda [*{of}* *SpeciesName*] = Inverse_Element_Volume [Lagged = *lagged*]

Scope

Aria Material

Summary

This model has been deprecated and removed from the code.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
lagged	integer	0

Nd_Table

Syntax

Mesh Lambda [*{of}* *SpeciesName*] = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Mesh Lambda [*{of}* *SpeciesName*] = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Mesh Lambda [*{of}* *SpeciesName*] = Scalar_String_Function F = *f*

Scope

Aria Material

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	"string"	–

User_Field

Syntax

Mesh Lambda [*{of}* *SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Mesh Lambda [*{of}* *SpeciesName*] = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Scope

Aria Material

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Mesh Lambda [*{of} SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	“string”	–
plugin_parameters	“string”	–

Mesh Poissons Ratio

Constant

Syntax

Mesh Poissons Ratio [*{of} SpeciesName*] = Constant *{nu | value}* = *nu*

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
nu	real	–

Copied

Syntax

Mesh Poissons Ratio [*{of}* *SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	“string”	–

Exponential

Syntax

Mesh Poissons Ratio [*{of}* *SpeciesName*] = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Scope

Aria Material

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Mesh Poissons Ratio [*{of} SpeciesName*] = Global (Global_Name = *global_name* | [Component = *component*])

Scope

Aria Material

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Mesh Poissons Ratio [*{of} SpeciesName*] = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Mesh Poissons Ratio [*{of}* *SpeciesName*] = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Mesh Poissons Ratio [*{of}* *SpeciesName*] = Scalar_String_Function F = *f*

Scope

Aria Material

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	"string"	–

User_Field

Syntax

Mesh Poissons Ratio [*{of} SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Mesh Poissons Ratio [*{of} SpeciesName*] = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Scope

Aria Material

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Mesh Poissons Ratio [*{of}* *SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	“string”	–
plugin_parameters	“string”	–

Mesh Stress

Constant

Syntax

Mesh Stress [*{of}* *SpeciesName*] = Constant ([Xx = *xx*] | [Xy = *xy*] | [Xz = *xz*] | [Yx = *yx*] | [Yy = *yy*] | [Yz = *yz*] | [Zx = *zx*] | [Zy = *zy*] | [Zz = *zz*] | [T11 = *t11*] | [T12 = *t12*] | [T13 = *t13*] | [T21 = *t21*] | [T22 = *t22*] | [T23 = *t23*] | [T31 = *t31*] | [T32 = *t32*] | [T33 = *t33*])

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
xx	real	0
xy	real	0
xz	real	0
yx	real	0
yy	real	0
yz	real	0
zx	real	0
zy	real	0
zz	real	0
t11	real	0
t12	real	0
t13	real	0
t21	real	0
t22	real	0
t23	real	0
t31	real	0
t32	real	0
t33	real	0

Electrode_Old

Syntax

Mesh Stress [*{of}* *SpeciesName*] = Electrode_Old (F = *f* | Electrodefile = *electrodeFile* | Kmolconversion = *kmolConversion* | [Jconversion = *JConversion*] | [Meterconversion = *meterConversion*] | [Max_Sub_Timesteps = *max_sub_timesteps*] | [Base_Delta = *base_delta*] | [Jam_Porosity = *jam_porosity*] | [Jam_Width = *jam_width*])

Scope

Aria Material

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	real	–
electrodeFile	“string”	–
kmolConversion	real	–
JConversion	real	1
meterConversion	real	1
max_sub_timesteps	integer	100
base_delta	real	1e-05
jam_porosity	real	0.25
jam_width	real	0.1

Gent_Hyperelastic

Syntax

Mesh Stress [*{of}* *SpeciesName*] = Gent_Hyperelastic [Jm = *Jm*]

Scope

Aria Material

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Jm	real	1

Isothermal

Syntax

Mesh Stress [*{of}* *SpeciesName*] = Isothermal (T = *t* \| T_Ref = *t_ref*)

Scope

Aria Material

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
t	real	–
t_ref	real	–

Lame

Syntax

Mesh Stress [*{of} SpeciesName*] = Lame Model = *model*

Scope

Aria Material

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
model	“string”	–

Linear_Elastic

Syntax

Mesh Stress [*{of} SpeciesName*] = Linear_Elastic [Reference_Frame = *reference_frame*]

Scope

Aria Material

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
reference_frame	“string”	MOVING

Mooney_Rivlin

Syntax

Mesh Stress [*{of} SpeciesName*] = Mooney_Rivlin ()

Scope

Aria Material

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–

Neohookean_Elastic

Syntax

Mesh Stress [*{of} SpeciesName*] = Neohookean_Elastic ()

Scope

Aria Material

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–

Nonlinear_Elastic

Syntax

Mesh Stress [*{of} SpeciesName*] = Nonlinear_Elastic ()

Scope

Aria Material

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–

Porous_Effective

Syntax

Mesh Stress [*{of} SpeciesName*] = Porous_Effective ([Pressure = *pressure*] | [Phase = *phase*] | [Biot = *biot*])

Scope

Aria Material

Summary

Apply the pressure as an isotropic mesh stress. Primarily used as an effective stress in porous flow problems.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
pressure	“string”	–
phase	“string”	–
biot	real	1

Remove_Old_Mesh_Stress

Syntax

Mesh Stress [*{of}* *SpeciesName*] = Remove_Old_Mesh_Stress ()

Scope

Aria Material

Summary

Subtracts off the total `MESH_STRESS_OLD` (not including contributions from this model)

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–

Residual

Syntax

Mesh Stress [*{of}* *SpeciesName*] = Residual ([Sx = *sx*] | [Sy = *sy*] | [Sz = *sz*] | [Sxx = *sxx*] | [Sxy = *sxy*] | [Sxz = *sxz*] | [Syy = *syy*] | [Syz = *syz*] | [Szz = *szz*])

Scope

Aria Material

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
sx	real	0
sy	real	0
sz	real	0
sxx	real	0
sxy	real	0
sxz	real	0
syy	real	0
syz	real	0
szz	real	0

Saturation_Weighted_Porous_Effective

Syntax

Mesh Stress [*{of} SpeciesName*] = Saturation_Weighted_Porous_Effective ([Pressure = *pressure*] | [Phase = *phase*] | [Biot = *biot*])

Scope

Aria Material

Summary

Apply the saturation-weighted pressure as an isotropic mesh stress. Primarily used as an effective stress in porous flow problems.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
pressure	"string"	–
phase	"string"	–
biot	real	1

Species

Syntax

Mesh Stress [*{of} SpeciesName*] = Species (Sref = *sref* | Phase = *phase* | Species = *species*)

Scope

Aria Material

Summary

A linear mechanical strain due to a species concentration

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
sref	real	–
phase	"string"	–
species	"string"	–

Species_Anisotropic

Syntax

Mesh Stress [*{of}* *SpeciesName*] = Species_Anisotropic (Sref = *sref* | [Phase = *phase*] | [Species = *species*])

Scope

Aria Material

Summary

Anisotropic linear mechanical strain due to a species concentration

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
sref	real	–
phase	"string"	–
species	"string"	–

Species_Transversely_Isotropic

Syntax

Mesh Stress [*{of}* *SpeciesName*] = Species_Transversely_Isotropic (Refval = *refval* | Beta_N = *beta_n* | Beta_T = *beta_t* | [N_X = *n_x*] | [N_Y = *n_y*] | [N_Z = *n_z*] | [Phase = *phase*] | [Species = *species*])

Scope

Aria Material

Summary

A linear mechanical strain that is transversely isotropic due to a species concentration

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
refval	real	–
beta_n	real	–
beta_t	real	–
n_x	real	0
n_y	real	0
n_z	real	0
phase	"string"	–
species	"string"	–

Tensor_String_Function

Syntax

Mesh Stress [*{of} SpeciesName*] = Tensor_String_Function (F_Xx = *f_xx* | F_Yy = *f_yy* | [F_Xy = *f_xy*] | [F_Zz = *f_zz*] | [F_Xz = *f_xz*] | [F_Yz = *f_yz*] | [F_Yx = *f_yx*] | [F_Zx = *f_zx*] | [F_Zy = *f_zy*])

Scope

Aria Material

Summary

Value from a string function.

Description

The tensor components are calculated from the user-defined string functions, which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used. Off-diagonal terms are optional and default to 0 if omitted.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f_xx	"string"	–
f_yy	"string"	–
f_xy	"string"	0.0
f_zz	"string"	–
f_xz	"string"	0.0
f_yz	"string"	0.0
f_yx	"string"	0.0
f_zx	"string"	0.0
f_zy	"string"	0.0

Tensor_String_Function_Symmetric

Syntax

Mesh Stress [*{of} SpeciesName*] = Tensor_String_Function_Symmetric (F_Xx = *f_xx* | F_Yy = *f_yy* | [F_Xy = *f_xy*] | [F_Zz = *f_zz*] | [F_Xz = *f_xz*] | [F_Yz = *f_yz*])

Scope

Aria Material

Summary

Value from a string function.

Description

The symmetric tensor components are calculated from the user-defined string functions (only for diagonal and upper triangular terms), which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used. Off-diagonal terms are optional and default to 0 if omitted.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f_xx	"string"	–
f_yy	"string"	–
f_xy	"string"	0.0
f_zz	"string"	–
f_xz	"string"	0.0
f_yz	"string"	0.0

Thermal

Syntax

Mesh Stress [*{of}* *SpeciesName*] = Thermal ([T_Ref = *T_ref*] | [Beta = *beta*])

Scope

Aria Material

Summary

A linear mechanical strain due to a temperature change from reference temperature.

$$\sigma_{xx} = \sigma_{yy} = \sigma_{zz} = beta(T - T_{ref})$$

where β is a thermal expansion coefficient and T_{ref} is some reference temperature. Alternatively, beta can also be specified as a model in the material

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
T_ref	real	0
beta	real	0

User_Plugin

Syntax

Mesh Stress [*{of} SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	“string”	–
plugin_parameters	“string”	–

Mesh Surface Tension

Constant

Syntax

Mesh Surface Tension [*{of} SpeciesName*] = Constant *{sigma | value}* = *sigma*

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
sigma	real	–

Copied

Syntax

Mesh Surface Tension [*{of}* *SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	"string"	–

Exponential

Syntax

Mesh Surface Tension [*{of}* *SpeciesName*] = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Scope

Aria Material

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Mesh Surface Tension [*{of}* *SpeciesName*] = Global (Global_Name = *global_name* | [Component = *component*])

Scope

Aria Material

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
global_name	"string"	–
component	integer	0

Linear_T

Syntax

Mesh Surface Tension [*{of}* *SpeciesName*] = Linear_T (Sigma0 = *sigma0* | Dsigmadt = *dsigmadT* | T_Ref = *T_ref*)

Scope

Aria Material

Summary

Linear T mesh surface tension

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
sigma0	real	–
dsigmadT	real	–
T_ref	real	–

Nd_Table

Syntax

Mesh Surface Tension [*{of} SpeciesName*] = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Mesh Surface Tension [*{of} SpeciesName*] = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Mesh Surface Tension [*{of} SpeciesName*] = Scalar_String_Function F = *f*

Scope

Aria Material

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	"string"	–

User_Field

Syntax

Mesh Surface Tension [*{of} SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Mesh Surface Tension [*{of} SpeciesName*] = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Scope

Aria Material

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Mesh Surface Tension [*{of} SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	“string”	–
plugin_parameters	“string”	–

Mesh Two Mu

Constant

Syntax

Mesh Two Mu [*{of} SpeciesName*] = Constant *{two_mu | value}* = *two_mu*

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
two_mu	real	–

Converted

Syntax

Mesh Two Mu [*{of}* *SpeciesName*] = Converted ()

Scope

Aria Material

Summary

Lame coefficient computed from Poisson's ratio and elastic modulus

Description

$$2\mu = \frac{E}{1+\nu}$$

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–

Copied

Syntax

Mesh Two Mu [*{of}* *SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	"string"	–

Elemental_Volume

Syntax

Mesh Two Mu [*{of}* *SpeciesName*] = Elemental_Volume ()

Scope

Aria Material

Summary

2 mu computed from elemental volume

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–

Exponential

Syntax

Mesh Two Mu [*{of}* *SpeciesName*] = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Scope

Aria Material

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	“string”	–
constant	real	0
multiplier	real	1
exponent	real	–

Faux_Plasticity

Syntax

Mesh Two Mu [*{of}* *SpeciesName*] = Faux_Plasticity (E = *e* | Yield_Stress = *yield_stress* | Hard = *hard* | [Min_Stress_Multiplier = *min_stress_multiplier*])

Scope

Aria Material

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
e	real	–
yield_stress	real	–
hard	"string"	–
min_stress_multiplier	real	0.01

Global

Syntax

Mesh Two Mu [*{of}* *SpeciesName*] = Global (Global_Name = *global_name* | [Component = *component*])

Scope

Aria Material

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
global_name	"string"	–
component	integer	0

Inverse_Element_Volume

Syntax

Mesh Two Mu [*{of}* *SpeciesName*] = Inverse_Element_Volume [Lagged = *lagged*]

Scope

Aria Material

Summary

This model has been deprecated and removed from the code.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
lagged	integer	0

Nd_Table

Syntax

Mesh Two Mu [*{of}* *SpeciesName*] = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Mesh Two Mu [*{of}* *SpeciesName*] = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Mesh Two Mu [*{of}* *SpeciesName*] = Scalar_String_Function F = *f*

Scope

Aria Material

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be

a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	"string"	–

User_Field

Syntax

Mesh Two Mu [*{of}* *SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Mesh Two Mu [*{of}* *SpeciesName*] = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Scope

Aria Material

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1
toggle	"string"	–

User_Plugin

Syntax

Mesh Two Mu [*{of}* *SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	"string"	–
plugin_parameters	"string"	–

Mesh Velocity

Constant

Syntax

Mesh Velocity [*{of}* *SpeciesName*] = Constant ([X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
x	real	0
y	real	0
z	real	0

Copied

Syntax

Mesh Velocity [*{of}* *SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	"string"	–

User_Field

Syntax

Mesh Velocity [*{of}* *SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Vector from User Field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Mesh Velocity [*{of}* *SpeciesName*] = User_Function ([Name_X = *name_x*] | [Name_Y = *name_y*] | [Name_Z = *name_z*] | X = *x* | [Multiplier = *multiplier*])

Scope

Aria Material

Summary

Value from a Vector User Function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name_x	"string"	–
name_y	"string"	–
name_z	"string"	–
x	"string"	–
multiplier	real	1

User_Plugin

Syntax

Mesh Velocity [*{of}* *SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	"string"	–
plugin_parameters	"string"	–

Vector_String_Function

Syntax

Mesh Velocity [*{of}* *SpeciesName*] = Vector_String_Function (F_X = *f_x* | F_Y = *f_y* | [F_Z = *f_z*])

Scope

Aria Material

Summary

Value from a string function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f_x	"string"	–
f_y	"string"	–
f_z	"string"	–

Mesh Velocity X

Constant

Syntax

Mesh Velocity X [*{of}* *SpeciesName*] = Constant Value = *value*

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
value	real	–

Copied

Syntax

Mesh Velocity X [*{of}* *SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	"string"	–

Exponential

Syntax

Mesh Velocity X [*{of}* *SpeciesName*] = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Scope

Aria Material

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Mesh Velocity X [*{of} SpeciesName*] = Global (Global_Name = *global_name* | [Component = *component*])

Scope

Aria Material

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Mesh Velocity X [*{of} SpeciesName*] = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
vars	"string"1["string"2["string"3["string"4["string"5]]]]	–
i0	"string"	–
i1	"string"	–
i2	"string"	–
x	"string"	–
y	"string"	–
z	"string"	–

Polynomial

Syntax

Mesh Velocity X [*{of}* *SpeciesName*] = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Mesh Velocity X [*{of}* *SpeciesName*] = Scalar_String_Function F = *f*

Scope

Aria Material

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be

a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	"string"	–

User_Field

Syntax

Mesh Velocity X [*{of}* *SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Mesh Velocity X [*{of}* *SpeciesName*] = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Scope

Aria Material

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Mesh Velocity X [*{of} SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	“string”	–
plugin_parameters	“string”	–

Mesh Velocity Y

Constant

Syntax

Mesh Velocity Y [*{of} SpeciesName*] = Constant Value = *value*

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
value	real	–

Copied

Syntax

Mesh Velocity Y [*{of}* *SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	“string”	–

Exponential

Syntax

Mesh Velocity Y [*{of}* *SpeciesName*] = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Scope

Aria Material

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Mesh Velocity Y [*{of}* *SpeciesName*] = Global (Global_Name = *global_name* | [Component = *component*])

Scope

Aria Material

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Mesh Velocity Y [*{of}* *SpeciesName*] = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Mesh Velocity Y [*{of}* *SpeciesName*] = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Mesh Velocity Y [*{of}* *SpeciesName*] = Scalar_String_Function F = *f*

Scope

Aria Material

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	"string"	–

User_Field

Syntax

Mesh Velocity Y [*{of}* *SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Mesh Velocity Y [*{of}* *SpeciesName*] = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Scope

Aria Material

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Mesh Velocity Y [*{of} SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	“string”	–
plugin_parameters	“string”	–

Mesh Velocity Z

Constant

Syntax

Mesh Velocity Z [*{of} SpeciesName*] = Constant Value = *value*

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
value	real	–

Copied

Syntax

Mesh Velocity Z [*{of}* *SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	"string"	–

Exponential

Syntax

Mesh Velocity Z [*{of}* *SpeciesName*] = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Scope

Aria Material

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Mesh Velocity Z [*{of} SpeciesName*] = Global (Global_Name = *global_name* | [Component = *component*])

Scope

Aria Material

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Mesh Velocity Z [*{of} SpeciesName*] = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Mesh Velocity Z [*{of}* *SpeciesName*] = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Mesh Velocity Z [*{of} SpeciesName*] = Scalar_String_Function F = *f*

Scope

Aria Material

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	"string"	–

User_Field

Syntax

Mesh Velocity Z [*{of} SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Mesh Velocity Z [*{of} SpeciesName*] = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Scope

Aria Material

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Mesh Velocity Z [*{of} SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	“string”	–
plugin_parameters	“string”	–

Mesh Youngs Modulus

Constant

Syntax

Mesh Youngs Modulus [*{of} SpeciesName*] = Constant *{e | value}* = *e*

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
e	real	–

Copied

Syntax

Mesh Youngs Modulus [*{of}* *SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	“string”	–

Exponential

Syntax

Mesh Youngs Modulus [*{of}* *SpeciesName*] = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Scope

Aria Material

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Faux_Plasticity

Syntax

Mesh Youngs Modulus [*{of}* *SpeciesName*] = Faux_Plasticity (E = *e* | Yield_Stress = *yield_stress* | Hard = *hard* | [Min_Stress_Multiplier = *min_stress_multiplier*])

Scope

Aria Material

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
e	real	–
yield_stress	real	–
hard	"string"	–
min_stress_multiplier	real	0.01

Global

Syntax

Mesh Youngs Modulus [*{of}* *SpeciesName*] = Global (Global_Name = *global_name* | [Component = *component*])

Scope

Aria Material

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
global_name	"string"	–
component	integer	0

Melting_Jamming_Porous

Syntax

Mesh Youngs Modulus [*{of}* *SpeciesName*] = Melting_Jamming_Porous (Jam_Porosity = *jam_porosity* | Jam_Width = *jam_width* | Solid_Modulus = *solid_modulus* | Melted_Modulus = *melted_modulus* | Jammed_Modulus = *jammed_modulus* | Melt_Dt = *melt_DT*)

Scope

Aria Material

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
jam_porosity	real	–
jam_width	real	–
solid_modulus	real	–
melted_modulus	real	–
jammed_modulus	real	–
melt_DT	real	–

Nd_Table

Syntax

Mesh Youngs Modulus [*{of}* *SpeciesName*] = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Mesh Youngs Modulus [*{of}* *SpeciesName*] = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Mesh Youngs Modulus [*{of}* *SpeciesName*] = Scalar_String_Function F = *f*

Scope

Aria Material

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	"string"	–

User_Field

Syntax

Mesh Youngs Modulus [*{of}* *SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Mesh Youngs Modulus [*{of}* *SpeciesName*] = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Scope

Aria Material

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Mesh Youngs Modulus [*{of}* *SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	“string”	–
plugin_parameters	“string”	–

Mixture Density

Constant

Syntax

Mixture Density [*{of}* *SpeciesName*] = Constant Value = *value*

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
value	real	–

Copied

Syntax

Mixture Density [*{of}* *SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	“string”	–

Exponential

Syntax

Mixture Density [*{of}* *SpeciesName*] = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Scope

Aria Material

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Mixture Density [*{of} SpeciesName*] = Global (Global_Name = *global_name* | [Component = *component*])

Scope

Aria Material

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Mixture Density [*{of} SpeciesName*] = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Mixture Density [*{of}* *SpeciesName*] = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Mixture Density [*{of}* *SpeciesName*] = Scalar_String_Function F = *f*

Scope

Aria Material

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	"string"	–

User_Field

Syntax

Mixture Density [*{of} SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Mixture Density [*{of} SpeciesName*] = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Scope

Aria Material

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Mixture Density [*{of}* *SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	“string”	–
plugin_parameters	“string”	–

Mixture Fraction

Constant

Syntax

Mixture Fraction [*{of}* *SpeciesName*] = Constant Value = *value*

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
value	real	–

Copied

Syntax

Mixture Fraction [*{of}* *SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	“string”	–

Exponential

Syntax

Mixture Fraction [*{of}* *SpeciesName*] = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Scope

Aria Material

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Mixture Fraction [*{of}* *SpeciesName*] = Global (Global_Name = *global_name* | [Component = *component*])

Scope

Aria Material

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Mixture Fraction [*{of}* *SpeciesName*] = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Mixture Fraction [*{of}* *SpeciesName*] = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Mixture Fraction [*{of} SpeciesName*] = Scalar_String_Function F = *f*

Scope

Aria Material

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	"string"	–

User_Field

Syntax

Mixture Fraction [*{of}* *SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Mixture Fraction [*{of}* *SpeciesName*] = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Scope

Aria Material

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Mixture Fraction [*{of}* *SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	“string”	–
plugin_parameters	“string”	–

Mixture Fraction Diffusive Flux

Basic

Syntax

Mixture Fraction Diffusive Flux [*{of}* *SpeciesName*] = Basic ()

Scope

Aria Material

Summary

Basic model of mixture fraction diffusive flux

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–

Mixture Fraction Diffusivity

Constant

Syntax

Mixture Fraction Diffusivity [*{of} SpeciesName*] = Constant *{dz | value}* = *Dz*

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Dz	real	–

Copied

Syntax

Mixture Fraction Diffusivity [*{of} SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	“string”	–

Exponential

Syntax

Mixture Fraction Diffusivity [*{of} SpeciesName*] = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Scope

Aria Material

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

From_Schmidt

Syntax

Mixture Fraction Diffusivity [*{of}* *SpeciesName*] = From_Schmidt ()

Scope

Aria Material

Summary

Mixture fraction diffusivity from Schmidt number and viscosity

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–

Global

Syntax

Mixture Fraction Diffusivity [*{of}* *SpeciesName*] = Global (Global_Name = *global_name* | [Component = *component*])

Scope

Aria Material

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
global_name	“string”	–
component	integer	0

Nd_Table

Syntax

Mixture Fraction Diffusivity [*{of} SpeciesName*] = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Mixture Fraction Diffusivity [*{of} SpeciesName*] = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Mixture Fraction Diffusivity [*{of}* *SpeciesName*] = Scalar_String_Function F = *f*

Scope

Aria Material

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	"string"	–

User_Field

Syntax

Mixture Fraction Diffusivity [*{of}* *SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Mixture Fraction Diffusivity [*{of}* *SpeciesName*] = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Scope

Aria Material

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1
toggle	"string"	–

User_Plugin

Syntax

Mixture Fraction Diffusivity [*{of} SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	"string"	–
plugin_parameters	"string"	–

Mixture Mass Diffusivity

Constant

Syntax

Mixture Mass Diffusivity [*{of} SpeciesName*] = Constant *{dmix | value}* = *Dmix*

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Dmix	real	–

Copied

Syntax

Mixture Mass Diffusivity [*{of}* *SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	“string”	–

Exponential

Syntax

Mixture Mass Diffusivity [*{of}* *SpeciesName*] = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Scope

Aria Material

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	“string”	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Mixture Mass Diffusivity [*{of}* *SpeciesName*] = Global (Global_Name = *global_name* | [Component = *component*])

Scope

Aria Material

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
global_name	“string”	–
component	integer	0

Mixture_Average

Syntax

Mixture Mass Diffusivity [*{of}* *SpeciesName*] = Mixture_Average ()

Scope

Aria Material

Summary

Linear mixture-averaged mass diffusivity

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–

Nd_Table

Syntax

Mixture Mass Diffusivity [*{of}* *SpeciesName*] = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Mixture Mass Diffusivity [*{of}* *SpeciesName*] = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Mixture Mass Diffusivity [*{of}* *SpeciesName*] = Scalar_String_Function F = *f*

Scope

Aria Material

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	"string"	–

User_Field

Syntax

Mixture Mass Diffusivity [*{of}* *SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Mixture Mass Diffusivity [*{of}* *SpeciesName*] = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Scope

Aria Material

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1
toggle	"string"	–

User_Plugin

Syntax

Mixture Mass Diffusivity [*{of}* *SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	"string"	–
plugin_parameters	"string"	–

Mixture Molecular Weight

Cantera

Syntax

Mixture Molecular Weight [*{of}* *SpeciesName*] = Cantera [Poffset = *poffset*]

Scope

Aria Material

Summary

Cantera mixture molecular weight

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
poffset	real	0

Constant

Syntax

Mixture Molecular Weight [*{of} SpeciesName*] = Constant *{mmix | value}* = *Mmix*

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Mmix	real	–

Copied

Syntax

Mixture Molecular Weight [*{of} SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	“string”	–

Exponential

Syntax

Mixture Molecular Weight [*{of}* *SpeciesName*] = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Scope

Aria Material

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

From_Material_Phase

Syntax

Mixture Molecular Weight [*{of}* *SpeciesName*] = From_Material_Phase Phase_Name = *phase_name*

Scope

Aria Material

Summary

Copy the mixture molecular weight from another material phase to the desired material phase.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
phase_name	"string"	–

Global

Syntax

Mixture Molecular Weight [*{of}* *SpeciesName*] = Global (Global_Name = *global_name* | [Component = *component*])

Scope

Aria Material

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
global_name	"string"	–
component	integer	0

Mass_Average

Syntax

Mixture Molecular Weight [*{of}* *SpeciesName*] = Mass_Average ()

Scope

Aria Material

Summary

Mass averaged mixture molecular weight

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–

Mole_Average

Syntax

Mixture Molecular Weight [*{of}* *SpeciesName*] = Mole_Average [Species_Variable = *species_variable*]

Scope

Aria Material

Summary

Mole averaged mixture molecular weight

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
species_variable	“string”	SPECIES_FRACTION

Nd_Table

Syntax

Mixture Molecular Weight [*{of} SpeciesName*] = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Mixture Molecular Weight [*{of} SpeciesName*] = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Mixture Molecular Weight [*{of}* *SpeciesName*] = Scalar_String_Function F = *f*

Scope

Aria Material

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	"string"	–

User_Field

Syntax

Mixture Molecular Weight [*{of}* *SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Mixture Molecular Weight [*{of}* *SpeciesName*] = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Scope

Aria Material

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1
toggle	"string"	–

User_Plugin

Syntax

Mixture Molecular Weight [*{of} SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	"string"	–
plugin_parameters	"string"	–

Molecular Weight

Cantera

Syntax

Molecular Weight [*{of} SpeciesName*] = Cantera [Poffset = *poffset*]

Scope

Aria Material

Summary

Cantera molecular weight

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
poffset	real	0

Constant

Syntax

Molecular Weight [*{of}* *SpeciesName*] = Constant *{m | value}* = *m*

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
m	real	–

Copied

Syntax

Molecular Weight [*{of}* *SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	“string”	–

Dynamic_Species

Syntax

Molecular Weight [*{of}* *SpeciesName*] = Dynamic_Species ()

Scope

Aria Material

Summary

Calculate the molecular weight dynamically from density/molar concentration.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–

Exponential

Syntax

Molecular Weight [*{of}* *SpeciesName*] = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Scope

Aria Material

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	“string”	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Molecular Weight [*{of}* *SpeciesName*] = Global (Global_Name = *global_name* | [Component = *component*])

Scope

Aria Material

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
global_name	“string”	–
component	integer	0

Nd_Table

Syntax

Molecular Weight [*{of}* *SpeciesName*] = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Molecular Weight [*{of}* *SpeciesName*] = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Molecular Weight [*{of}* *SpeciesName*] = Scalar_String_Function F = *f*

Scope

Aria Material

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be

a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	"string"	–

User_Field

Syntax

Molecular Weight [*{of}* *SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Molecular Weight [*{of}* *SpeciesName*] = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Scope

Aria Material

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1
toggle	"string"	–

User_Plugin

Syntax

Molecular Weight [*{of} SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	"string"	–
plugin_parameters	"string"	–

Momentum Face Stabilization Scaling

Default

Syntax

Momentum Face Stabilization Scaling ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) = Default ()

Scope

Aria Material

Summary

Default face stabilization scaling for momentum equation

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–

Momentum Stress

Bad_Sensitivity

Syntax

Momentum Stress [*{of}* *SpeciesName*] = Bad_Sensitivity ()

Scope

Aria Material

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–

Balanced_Force

Syntax

Momentum Stress [*{of}* *SpeciesName*] = Balanced_Force ()

Scope

Aria Material

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–

Formal_Newtonian

Syntax

Momentum Stress [*{of}* *SpeciesName*] = Formal_Newtonian ([Pressure = *pressure*] | [Phase = *phase*] | [Biot = *biot*] | [Reference_Frame = *reference_frame*])

Scope

Aria Material

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
pressure	“string”	–
phase	“string”	–
biot	real	1
reference_frame	“string”	MOVING

Grad_Div

Syntax

Momentum Stress [*{of}* *SpeciesName*] = Grad_Div ([Pressure = *pressure*] | [Multiplier = *multiplier*])

Scope

Aria Material

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
pressure	“string”	–
multiplier	real	1

Incompressible_Newtonian

Syntax

Momentum Stress [*{of}* *SpeciesName*] = Incompressible_Newtonian ([Pressure = *pressure*] | [Phase = *phase*] | [Biot = *biot*] | [Reference_Frame = *reference_frame*])

Scope

Aria Material

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
pressure	“string”	–
phase	“string”	–
biot	real	1
reference_frame	“string”	MOVING

Lorentz

Syntax

Momentum Stress [*{of}* *SpeciesName*] = Lorentz ()

Scope

Aria Material

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–

Ls_Capillary

Syntax

Momentum Stress [*{of}* *SpeciesName*] = Ls_Capillary ()

Scope

Aria Material

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–

Maxwell

Syntax

Momentum Stress [*{of}* *SpeciesName*] = Maxwell ()

Scope

Aria Material

Summary

The gradient of the Maxwell stress tensor is the force on a fluid due to an

electrical field and is non-zero when there is either a net charge density or a non-uniform electric permittivity. $\nabla \cdot \mathbf{M} = \rho_{charge}\mathbf{E} - \frac{1}{2}(\mathbf{E} \cdot \mathbf{E})\nabla\epsilon$

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–

Mckenzie

Syntax

Momentum Stress [*{of}* *SpeciesName*] = Mckenzie ()

Scope

Aria Material

Summary

Provides a stress tensor based on a spherical and deviatoric decomposition. The pressure is derived from the pressure of the gas phase and the divergence of the flow field. The deviatoric stress is linear in the deviatoric velocity gradient.$\sigma = (1. - \phi)((P_g - \beta\nabla \cdot v)I + \frac{\mu}{2}(\nabla v + \nabla v^T - \frac{2}{3}\nabla \cdot v))$

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–

Newtonian_Dilational

Syntax

Momentum Stress [*{of}* *SpeciesName*] = Newtonian_Dilational [Reference_Frame = *reference_frame*]

Scope

Aria Material

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
reference_frame	“string”	MOVING

Newtonian_Pressure

Syntax

Momentum Stress [*{of}* *SpeciesName*] = Newtonian_Pressure ([Pressure = *pressure*] | [Phase = *phase*] | [Biot = *biot*])

Scope

Aria Material

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
pressure	"string"	–
phase	"string"	–
biot	real	1

Newtonian_Viscous

Syntax

Momentum Stress [*{of}* *SpeciesName*] = Newtonian_Viscous [Reference_Frame = *reference_frame*]

Scope

Aria Material

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
reference_frame	"string"	MOVING

Suspension

Syntax

Momentum Stress [*{of}* *SpeciesName*] = Suspension ([Pressure = *pressure*] | [Phase = *phase*] | [Biot = *biot*] | [Reference_Frame = *reference_frame*])

Scope

Aria Material

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
pressure	“string”	–
phase	“string”	NO_MATERIAL_PHASE
biot	real	1
reference_frame	“string”	MOVING

N

Neutron

Constant

Syntax

Neutron [*{of}* *SpeciesName*] = Constant Value = *value*

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
value	real	–

Copied

Syntax

Neutron [*{of}* *SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	"string"	–

Exponential

Syntax

Neutron [*{of}* *SpeciesName*] = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Scope

Aria Material

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Neutron [*{of}* *SpeciesName*] = Global (Global_Name = *global_name* | [Component = *component*])

Scope

Aria Material

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
global_name	“string”	–
component	integer	0

Nd_Table

Syntax

Neutron [*{of}* *SpeciesName*] = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Neutron [*{of}* *SpeciesName*] = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Neutron [*{of}* *SpeciesName*] = Scalar_String_Function F = *f*

Scope

Aria Material

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	"string"	–

User_Field

Syntax

Neutron [*{of} SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Neutron [*{of} SpeciesName*] = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Scope

Aria Material

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Neutron [*{of}* *SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	“string”	–
plugin_parameters	“string”	–

Neutron Diffusion

Basic

Syntax

Neutron Diffusion [*{of}* *SpeciesName*] = Basic ()

Scope

Aria Material

Summary

Basic neutron diffusion

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–

Ficks_Law

Syntax

Neutron Diffusion [*{of} SpeciesName*] = Ficks_Law ()

Scope

Aria Material

Summary

Fick's Law neutron diffusion

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–

Neutron Diffusivity

Constant

Syntax

Neutron Diffusivity [*{of} SpeciesName*] = Constant *{d | value}* = *d*

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
d	real	–

Copied

Syntax

Neutron Diffusivity [*{of} SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	"string"	–

Exponential

Syntax

Neutron Diffusivity [*{of}* *SpeciesName*] = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Scope

Aria Material

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Five_Ten

Syntax

Neutron Diffusivity [*{of}* *SpeciesName*] = Five_Ten ()

Scope

Aria Material

Summary

Five ten neutron diffusivity

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–

Global

Syntax

Neutron Diffusivity [*{of}* *SpeciesName*] = Global (Global_Name = *global_name* | [Component = *component*])

Scope

Aria Material

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Neutron Diffusivity [*{of}* *SpeciesName*] = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Neutron Diffusivity [*{of}* *SpeciesName*] = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Neutron Diffusivity [*{of}* *SpeciesName*] = Scalar_String_Function F = *f*

Scope

Aria Material

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	"string"	–

User_Field

Syntax

Neutron Diffusivity [*{of}* *SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Neutron Diffusivity [*{of}* *SpeciesName*] = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Scope

Aria Material

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Neutron Diffusivity [*{of} SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	“string”	–
plugin_parameters	“string”	–

Nonwetting Phase Retardation

Constant

Syntax

Nonwetting Phase Retardation [*{of} SpeciesName*] = Constant *{r | value}* = *r*

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
r	real	–

Copied

Syntax

Nonwetting Phase Retardation [*{of}* *SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	“string”	–

Exponential

Syntax

Nonwetting Phase Retardation [*{of}* *SpeciesName*] = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Scope

Aria Material

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Nonwetting Phase Retardation [*{of} SpeciesName*] = Global (Global_Name = *global_name* | [Component = *component*])

Scope

Aria Material

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Nonwetting Phase Retardation [*{of} SpeciesName*] = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Nonwetting Phase Retardation [*{of}* *SpeciesName*] = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Nonwetting Phase Retardation [*{of} SpeciesName*] = Scalar_String_Function F = *f*

Scope

Aria Material

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	"string"	–

User_Field

Syntax

Nonwetting Phase Retardation [*{of} SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Nonwetting Phase Retardation [*{of} SpeciesName*] = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Scope

Aria Material

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Nonwetting Phase Retardation [*{of} SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	“string”	–
plugin_parameters	“string”	–

Normal Mesh Velocity

Constant

Syntax

Normal Mesh Velocity [*{of} SpeciesName*] = Constant *{v | value}* = *v*

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
v	real	–

Copied

Syntax

Normal Mesh Velocity [*{of}* *SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	“string”	–

Exponential

Syntax

Normal Mesh Velocity [*{of}* *SpeciesName*] = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Scope

Aria Material

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Normal Mesh Velocity [*{of}* *SpeciesName*] = Global (Global_Name = *global_name* | [Component = *component*])

Scope

Aria Material

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Normal Mesh Velocity [*{of}* *SpeciesName*] = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Normal Mesh Velocity [*{of}* *SpeciesName*] = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Normal Mesh Velocity [*{of}* *SpeciesName*] = Scalar_String_Function F = *f*

Scope

Aria Material

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	"string"	–

User_Field

Syntax

Normal Mesh Velocity [*{of}* *SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Normal Mesh Velocity [*{of}* *SpeciesName*] = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Scope

Aria Material

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1
toggle	"string"	–

User_Plugin

Syntax

Normal Mesh Velocity [*{of} SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	"string"	–
plugin_parameters	"string"	–

O

Offset Darcy Potential

Constant

Syntax

Offset Darcy Potential ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) = Constant (Offset = *offset* | Surface = *surface* | [Equations = *Equations. . .*])

Scope

Aria Material

Summary

Offset DARCY_POTENTIAL by constant value on surfaces p' = p + offset.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
offset	real	–
surface	"string"	–
Equations	string. . .	–

Scalar_String_Function

Syntax

Offset Darcy Potential ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) = Scalar_String_Function (Offset = *offset* | Surface = *surface* | [Equations = *Equations. . .*])

Scope

Aria Material

Summary

Offset DARCY_POTENTIAL by string function on surfaces p' = p + offset.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
offset	"string"	–
surface	"string"	–
Equations	string. . .	–

Offset Mesh Displacements

Constant

Syntax

Offset Mesh Displacements ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) = Constant ([Offset = *offset1 offset2[offset3]*] | [Scale = *scale*] | Surface = *surface* | [Equations = *Equations. . .*])

Scope

Aria Material

Summary

Offset MESH_DISPLACEMENTS by constant value on surfaces d' = scale*d + <offset_x,offset_y,offset_z>. If a scale is provided, both the quantity and its flux are scaled accordingly.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
offset	real1 real2[real3]	0
scale	real	1
surface	"string"	–
Equations	string. . .	–

Reflected

Syntax

Offset Mesh Displacements ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) = Reflected ([Normal = *normal1 normal2[normal3]*] | [Scale = *scale*] | Surface = *surface* | [Equations = *Equations. . .*])

Scope

Aria Material

Summary

Reflects MESH_DISPLACEMENTS along plane given by normal. If a scale is provided (default is -1), both the quantity and its flux are scaled accordingly along normal.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
normal	real1 real2[real3]	0
scale	real	-1
surface	"string"	–
Equations	string. . .	–

Vector_String_Function

Syntax

Offset Mesh Displacements ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) = Vector_String_Function ([Offset_X = *offset_x*] | [Offset_Y = *offset_y*] | [Offset_Z = *offset_z*] | [Scale = *scale*] | Surface = *surface* | [Equations = *Equations. . .*])

Scope

Aria Material

Summary

Offset MESH_DISPLACEMENTS by string function on surfaces d' = scale*d + <offset_x,offset_y,offset_z>. If a scale is provided, both the quantity and its flux are scaled accordingly.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
offset_x	"string"	–
offset_y	"string"	–
offset_z	"string"	–
scale	real	1
surface	"string"	–
Equations	string. . .	–

Offset Pressure

Constant

Syntax

Offset Pressure ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) = Constant (Offset = *offset* | Surface = *surface* | [Equations = *Equations. . .*])

Scope

Aria Material

Summary

Offset PRESSURE by constant value on surfaces p' = p + offset.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
offset	real	–
surface	"string"	–
Equations	string. . .	–

Scalar_String_Function

Syntax

Offset Pressure ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) = Scalar_String_Function (Offset = *offset* | Surface = *surface* | [Equations = *Equations. . .*])

Scope

Aria Material

Summary

Offset PRESSURE by string function on surfaces p' = p + offset.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
offset	"string"	–
surface	"string"	–
Equations	string. . .	–

Offset Solid Displacements

Constant

Syntax

Offset Solid Displacements ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) = Constant ([Offset = *offset1 offset2[offset3]*] | [Scale = *scale*] | Surface = *surface* | [Equations = *Equations. . .*])

Scope

Aria Material

Summary

Offset SOLID_DISPLACEMENTS by constant value on surfaces d' = scale*d + <offset_x,offset_y,offset_z>. If a scale is provided, both the quantity and its flux are scaled accordingly.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
offset	real1 real2[real3]	0
scale	real	1
surface	"string"	–
Equations	string. . .	–

Reflected

Syntax

Offset Solid Displacements ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) = Reflected ([Normal = *normal1 normal2[normal3]*] | [Scale = *scale*] | Surface = *surface* | [Equations = *Equations. . .*])

Scope

Aria Material

Summary

Reflects SOLID_DISPLACEMENTS along plane given by normal. If a scale is provided (default is -1), both the quantity and its flux are scaled accordingly along normal.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
normal	real1 real2[real3]	0
scale	real	-1
surface	"string"	–
Equations	string. . .	–

Vector_String_Function

Syntax

Offset Solid Displacements ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) = Vector_String_Function ([Offset_X = *offset_x*] | [Offset_Y = *offset_y*] | [Offset_Z = *offset_z*] | [Scale = *scale*] | Surface = *surface* | [Equations = *Equations. . .*])

Scope

Aria Material

Summary

Offset SOLID_DISPLACEMENTS by string function on surfaces d' = scale*d + <offset_x,offset_y,offset_z>. If a scale is provided, both the quantity and its flux are scaled accordingly.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
offset_x	"string"	–
offset_y	"string"	–
offset_z	"string"	–
scale	real	1
surface	"string"	–
Equations	string. . .	–

Offset Species

Constant

Syntax

Offset Species ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) = Constant (Offset = *offset* | Surface = *surface* | [Equations = *Equations. . .*])

Scope

Aria Material

Summary

Offset SPECIES by constant value on surfaces s' = s + offset.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
offset	real	–
surface	"string"	–
Equations	string. . .	–

Scalar_String_Function

Syntax

Offset Species ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) = Scalar_String_Function (Offset = *offset* | Surface = *surface* | [Equations = *Equations. . .*])

Scope

Aria Material

Summary

Offset SPECIES by string function on surfaces s' = s + offset.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
offset	"string"	–
surface	"string"	–
Equations	string. . .	–

Offset Temperature

Constant

Syntax

Offset Temperature ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) = Constant (Offset = *offset* | Surface = *surface* | [Equations = *Equations. . .*])

Scope

Aria Material

Summary

Offset TEMPERATURE by constant value on surfaces T' = T + offset.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
offset	real	–
surface	"string"	–
Equations	string. . .	–

Scalar_String_Function

Syntax

Offset Temperature ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) = Scalar_String_Function (Offset = *offset* | Surface = *surface* | [Equations = *Equations. . .*])

Scope

Aria Material

Summary

Offset TEMPERATURE by string function on surfaces T' = T + offset.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
offset	"string"	–
surface	"string"	–
Equations	string. . .	–

Offset Velocity

Constant

Syntax

Offset Velocity ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) = Constant ([Offset = *offset1 offset2[offset3]*] | [Scale = *scale*] | Surface = *surface* | [Equations = *Equations. . .*])

Scope

Aria Material

Summary

Offset VELOCITY by constant value on surfaces v' = scale*v + <offset_x,offset_y,offset_z>. If a scale is provided, both the quantity and its flux are scaled accordingly.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
offset	real1 real2[real3]	0
scale	real	1
surface	"string"	–
Equations	string. . .	–

Reflected

Syntax

Offset Velocity ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) = Reflected ([Normal = *normal1 normal2[normal3]*] | [Scale = *scale*] | Surface = *surface* | [Equations = *Equations. . .*])

Scope

Aria Material

Summary

Reflects VELOCITY along plane given by normal. If a scale is provided (default is -1), both the quantity and its flux are scaled accordingly along normal.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
normal	real1 real2[real3]	0
scale	real	-1
surface	"string"	–
Equations	string. . .	–

Vector_String_Function

Syntax

Offset Velocity ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) = Vector_String_Function ([Offset_X = *offset_x*] | [Offset_Y = *offset_y*] | [Offset_Z = *offset_z*] | [Scale = *scale*] | Surface = *surface* | [Equations = *Equations. . .*])

Scope

Aria Material

Summary

Offset VELOCITY by string function on surfaces v' = scale*v + <offset_x,offset_y,offset_z>. If a scale is provided, both the quantity and its flux are scaled accordingly.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
offset_x	"string"	–
offset_y	"string"	–
offset_z	"string"	–
scale	real	1
surface	"string"	–
Equations	string. . .	–

Offset Voltage

Constant

Syntax

Offset Voltage ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) = Constant (Offset = *offset* | Surface = *surface* | [Equations = *Equations. . .*])

Scope

Aria Material

Summary

Offset VOLTAGE by constant value on surfaces V' = V + offset.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
offset	real	–
surface	"string"	–
Equations	string. . .	–

Scalar_String_Function

Syntax

Offset Voltage ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) = Scalar_String_Function (Offset = *offset* | Surface = *surface* | [Equations = *Equations. . .*])

Scope

Aria Material

Summary

Offset VOLTAGE by string function on surfaces V' = V + offset.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
offset	"string"	–
surface	"string"	–
Equations	string. . .	–

Open Circuit Voltage

Constant

Syntax

Open Circuit Voltage [*{of}* *SpeciesName*] = Constant ([Vx = *vx*] | [Vy = *vy*] | [Vz = *vz*])

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
vx	real	0
vy	real	0
vz	real	0

Copied

Syntax

Open Circuit Voltage [*{of}* *SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	“string”	–

Electrode_Object

Syntax

Open Circuit Voltage [*{of}* *SpeciesName*] = Electrode_Object ()

Scope

Aria Material

Summary

Open circuit voltage from the electrode

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–

Scalar_Vector_Product

Syntax

Open Circuit Voltage [*{of}* *SpeciesName*] = Scalar_Vector_Product (Vector = *vector* | [Scalars = *scalars*] | [Multiplier = *multiplier*])

Scope

Aria Material

Summary

Scalar vector product

Description

Multiplies a vector by a list of provided scalars (variables) and a multiplier (constant)

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
vector	“string”	–
scalars	“string”	–
multiplier	real	1

Tensor_Vector_Product

Syntax

Open Circuit Voltage [*{of}* *SpeciesName*] = Tensor_Vector_Product (Vector = *vector* | Tensor = *tensor* | [Scale = *scale*])

Scope

Aria Material

Summary

Tensor vector product

Description

Calculates a vector resulting from a tensor-vector product with an optional scaling factor.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
vector	"string"	–
tensor	"string"	–
scale	real	1

User_Field

Syntax

Open Circuit Voltage [*{of}* *SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Vector from User Field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Open Circuit Voltage [*{of} SpeciesName*] = User_Function ([Name_X = *name_x*] | [Name_Y = *name_y*] | [Name_Z = *name_z*] | X = *x* | [Multiplier = *multiplier*])

Scope

Aria Material

Summary

Value from a Vector User Function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name_x	"string"	–
name_y	"string"	–
name_z	"string"	–
x	"string"	–
multiplier	real	1

User_Plugin

Syntax

Open Circuit Voltage [*{of} SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	"string"	–
plugin_parameters	"string"	–

Vector_String_Function

Syntax

Open Circuit Voltage [*{of} SpeciesName*] = Vector_String_Function (F_X = *f_x* | F_Y = *f_y* | [F_Z = *f_z*])

Scope

Aria Material

Summary

Value from a string function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f_x	"string"	–
f_y	"string"	–
f_z	"string"	–

P

Partial Enclosure Area

Calore_User_Sub

Syntax

Partial Enclosure Area [*{of} SpeciesName*] = Calore_User_Sub (Name = *name* | Type = *type* | [Multiplier = *multiplier*] | [Material_Data_Block = *material_data_block*] | [Data = *data*] | [Scaling_Field = *scaling_field*])

Scope

Aria Material

Summary

Values from a Calore user subroutine

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
type	“string”	–
multiplier	real	1
material_data_block	“string”	–
data	“string”	–
scaling_field	“string”	–

Constant

Syntax

Partial Enclosure Area [*{of} SpeciesName*] = Constant *{a | value}* = *a*

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
a	real	–

Copied

Syntax

Partial Enclosure Area [*{of} SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	“string”	–

Exponential

Syntax

Partial Enclosure Area [*{of}* *SpeciesName*] = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Scope

Aria Material

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Partial Enclosure Area [*{of}* *SpeciesName*] = Global (Global_Name = *global_name* | [Component = *component*])

Scope

Aria Material

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Partial Enclosure Area [*{of}* *SpeciesName*] = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Partial Enclosure Area [*{of}* *SpeciesName*] = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Partial Enclosure Area [*{of}* *SpeciesName*] = Scalar_String_Function F = *f*

Scope

Aria Material

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	"string"	–

User_Field

Syntax

Partial Enclosure Area [*{of}* *SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Partial Enclosure Area [*{of}* *SpeciesName*] = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Scope

Aria Material

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Partial Enclosure Area [*{of} SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	“string”	–
plugin_parameters	“string”	–

Partial Enclosure Emissivity

Calore_User_Sub

Syntax

Partial Enclosure Emissivity [*{of} SpeciesName*] = Calore_User_Sub (Name = *name* | Type = *type* | [Multiplier = *multiplier*] | [Material_Data_Block = *material_data_block*] | [Data = *data*] | [Scaling_Field = *scaling_field*])

Scope

Aria Material

Summary

Values from a Calore user subroutine

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
type	“string”	–
multiplier	real	1
material_data_block	“string”	–
data	“string”	–
scaling_field	“string”	–

Constant

Syntax

Partial Enclosure Emissivity [*{of} SpeciesName*] = Constant *{e | value}* = *e*

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
e	real	–

Copied

Syntax

Partial Enclosure Emissivity [*{of} SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	“string”	–

Exponential

Syntax

Partial Enclosure Emissivity [*{of} SpeciesName*] = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Scope

Aria Material

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Partial Enclosure Emissivity [*{of} SpeciesName*] = Global (Global_Name = *global_name* | [Component = *component*])

Scope

Aria Material

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Partial Enclosure Emissivity [*{of}* *SpeciesName*] = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Partial Enclosure Emissivity [*{of}* *SpeciesName*] = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Partial Enclosure Emissivity [*{of} SpeciesName*] = Scalar_String_Function F = *f*

Scope

Aria Material

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	"string"	–

User_Field

Syntax

Partial Enclosure Emissivity [*{of} SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Partial Enclosure Emissivity [*{of} SpeciesName*] = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Scope

Aria Material

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Partial Enclosure Emissivity [*{of}* *SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	“string”	–
plugin_parameters	“string”	–

Partial Enclosure Temperature

Calore_User_Sub

Syntax

Partial Enclosure Temperature [*{of}* *SpeciesName*] = Calore_User_Sub (Name = *name* | Type = *type* | [Multiplier = *multiplier*] | [Material_Data_Block = *material_data_block*] | [Data = *data*] | [Scaling_Field = *scaling_field*])

Scope

Aria Material

Summary

Values from a Calore user subroutine

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
type	“string”	–
multiplier	real	1
material_data_block	“string”	–
data	“string”	–
scaling_field	“string”	–

Constant

Syntax

Partial Enclosure Temperature [*{of}* *SpeciesName*] = Constant *{t | value}* = *t*

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
t	real	–

Copied

Syntax

Partial Enclosure Temperature [*{of}* *SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	“string”	–

Exponential

Syntax

Partial Enclosure Temperature [*{of}* *SpeciesName*] = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Scope

Aria Material

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Partial Enclosure Temperature [*{of}* *SpeciesName*] = Global (Global_Name = *global_name* | [Component = *component*])

Scope

Aria Material

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Partial Enclosure Temperature [*{of}* *SpeciesName*] = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Partial Enclosure Temperature [*{of}* *SpeciesName*] = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Partial Enclosure Temperature [*{of}* *SpeciesName*] = Scalar_String_Function F = *f*

Scope

Aria Material

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	"string"	–

User_Field

Syntax

Partial Enclosure Temperature [*{of}* *SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Partial Enclosure Temperature [*{of}* *SpeciesName*] = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Scope

Aria Material

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Partial Enclosure Temperature [*{of} SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	“string”	–
plugin_parameters	“string”	–

Partial Molar Volume

Constant

Syntax

Partial Molar Volume [*{of} SpeciesName*] = Constant Value = *value*

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
value	real	–

Copied

Syntax

Partial Molar Volume [*{of}* *SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	“string”	–

Exponential

Syntax

Partial Molar Volume [*{of}* *SpeciesName*] = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Scope

Aria Material

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Partial Molar Volume [*{of}* *SpeciesName*] = Global (Global_Name = *global_name* | [Component = *component*])

Scope

Aria Material

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Partial Molar Volume [*{of}* *SpeciesName*] = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Partial Molar Volume [*{of}* *SpeciesName*] = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Partial Molar Volume [*{of} SpeciesName*] = Scalar_String_Function F = *f*

Scope

Aria Material

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	"string"	–

User_Field

Syntax

Partial Molar Volume [*{of}* *SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Partial Molar Volume [*{of}* *SpeciesName*] = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Scope

Aria Material

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Partial Molar Volume [*{of}* *SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	“string”	–
plugin_parameters	“string”	–

Peltier Coefficient

Constant

Syntax

Peltier Coefficient [*{of}* *SpeciesName*] = Constant Value = *value*

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
value	real	–

Copied

Syntax

Peltier Coefficient [*{of}* *SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	“string”	–

Exponential

Syntax

Peltier Coefficient [*{of}* *SpeciesName*] = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Scope

Aria Material

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Peltier Coefficient [*{of} SpeciesName*] = Global (Global_Name = *global_name* | [Component = *component*])

Scope

Aria Material

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Peltier Coefficient [*{of} SpeciesName*] = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Peltier Coefficient [*{of}* *SpeciesName*] = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Peltier Coefficient [*{of}* *SpeciesName*] = Scalar_String_Function F = *f*

Scope

Aria Material

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	"string"	–

User_Field

Syntax

Peltier Coefficient [*{of}* *SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Peltier Coefficient [*{of}* *SpeciesName*] = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Scope

Aria Material

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Peltier Coefficient [*{of} SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	“string”	–
plugin_parameters	“string”	–

Phase Change Specific Heat

Calore_User_Sub

Syntax

Phase Change Specific Heat [*{of} SpeciesName*] = Calore_User_Sub (Name = *name* | Type = *type* | [Multiplier = *multiplier*] | [Material_Data_Block = *material_data_block*] | [Data = *data*] | [Scaling_Field = *scaling_field*])

Scope

Aria Material

Summary

Values from a Calore user subroutine

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
type	“string”	–
multiplier	real	1
material_data_block	“string”	–
data	“string”	–
scaling_field	“string”	–

Constant

Syntax

Phase Change Specific Heat [*{of}* *SpeciesName*] = Constant *{cp | value}* = *Cp*

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Cp	real	–

Curing_Foam

Syntax

Phase Change Specific Heat [*{of}* *SpeciesName*] = Curing_Foam ()

Scope

Aria Material

Summary

Phase change specific heat for a curing foam

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–

Exponential

Syntax

Phase Change Specific Heat [*{of}* *SpeciesName*] = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Scope

Aria Material

Summary

Value from an exponential function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Phase Change Specific Heat [*{of}* *SpeciesName*] = Global (Global_Name = *global_name* | [Component = *component*])

Scope

Aria Material

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
global_name	"string"	–
component	integer	0

Polynomial

Syntax

Phase Change Specific Heat [*{of}* *SpeciesName*] = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Scope

Aria Material

Summary

Value from a polynomial function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	“string”	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Phase Change Specific Heat [*{of}* *SpeciesName*] = Scalar_String_Function F = *f*

Scope

Aria Material

Summary

Value from a string function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	"string"	–

User_Field

Syntax

Phase Change Specific Heat [*{of}* *SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Phase Change Specific Heat [*{of}* *SpeciesName*] = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Scope

Aria Material

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

Phi

Constant

Syntax

Phi [*{of}* *SpeciesName*] = Constant Value = *value*

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
value	real	–

Copied

Syntax

Phi [*{of}* *SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	"string"	–

Exponential

Syntax

Phi [*{of}* *SpeciesName*] = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Scope

Aria Material

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Phi [*{of}* *SpeciesName*] = Global (Global_Name = *global_name* | [Component = *component*])

Scope

Aria Material

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
global_name	“string”	–
component	integer	0

Nd_Table

Syntax

Phi [*{of} SpeciesName*] = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Phi [*{of} SpeciesName*] = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Phi [*{of}* *SpeciesName*] = Scalar_String_Function F = *f*

Scope

Aria Material

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	"string"	–

User_Field

Syntax

Phi [*{of}* *SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Phi [*{of}* *SpeciesName*] = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Scope

Aria Material

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Phi [*{of}* *SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	“string”	–
plugin_parameters	“string”	–

Plastic Strain

Constant

Syntax

Plastic Strain [*{of}* *SpeciesName*] = Constant Value = *value*

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
value	real	–

Copied

Syntax

Plastic Strain [*{of}* *SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	"string"	–

Exponential

Syntax

Plastic Strain [*{of}* *SpeciesName*] = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Scope

Aria Material

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Plastic Strain [*{of} SpeciesName*] = Global (Global_Name = *global_name* | [Component = *component*])

Scope

Aria Material

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Plastic Strain [*{of} SpeciesName*] = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
vars	"string"1["string"2["string"3["string"4["string"5]]]]	–
i0	"string"	–
i1	"string"	–
i2	"string"	–
x	"string"	–
y	"string"	–
z	"string"	–

Polynomial

Syntax

Plastic Strain [*{of}* *SpeciesName*] = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Plastic Strain [*{of}* *SpeciesName*] = Scalar_String_Function F = *f*

Scope

Aria Material

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be

a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	"string"	–

User_Field

Syntax

Plastic Strain [*{of}* *SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Plastic Strain [*{of}* *SpeciesName*] = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Scope

Aria Material

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Plastic Strain [*{of} SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	“string”	–
plugin_parameters	“string”	–

Polymerization Shift Factor

Constant

Syntax

Polymerization Shift Factor [*{of} SpeciesName*] = Constant Value = *value*

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
value	real	–

Copied

Syntax

Polymerization Shift Factor [*{of}* *SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	“string”	–

Exponential

Syntax

Polymerization Shift Factor [*{of}* *SpeciesName*] = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Scope

Aria Material

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Polymerization Shift Factor [*{of} SpeciesName*] = Global (Global_Name = *global_name* | [Component = *component*])

Scope

Aria Material

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Polymerization Shift Factor [*{of} SpeciesName*] = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Polymerization Shift Factor [*{of}* *SpeciesName*] = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Polymerization Shift Factor [*{of}* *SpeciesName*] = Scalar_String_Function F = *f*

Scope

Aria Material

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	"string"	–

User_Field

Syntax

Polymerization Shift Factor [*{of}* *SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Polymerization Shift Factor [*{of}* *SpeciesName*] = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Scope

Aria Material

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1
toggle	"string"	–

User_Plugin

Syntax

Polymerization Shift Factor [*{of}* *SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	"string"	–
plugin_parameters	"string"	–

Wlf

Syntax

Polymerization Shift Factor [*{of}* *SpeciesName*] = Wlf (C1 = *c1* | C2 = *c2*)

Scope

Aria Material

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
c1	real	–
c2	real	–

Pore Defect Radius

Constant

Syntax

Pore Defect Radius [*{of}* *SpeciesName*] = Constant Value = *value*

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
value	real	–

Copied

Syntax

Pore Defect Radius [*{of}* *SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	“string”	–

Exponential

Syntax

Pore Defect Radius [*{of}* *SpeciesName*] = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Scope

Aria Material

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Pore Defect Radius [*{of}* *SpeciesName*] = Global (Global_Name = *global_name* | [Component = *component*])

Scope

Aria Material

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
global_name	"string"	–
component	integer	0

Mmp_Model

Syntax

Pore Defect Radius [*{of}* *SpeciesName*] = Mmp_Model ([Beta0 = *beta0*] | [Beta1 = *beta1*] | B_Init = *b_init* | T_Init = *t_init* | [Bulk_Modulus = *BULK_MODUlUS*] | [Poisson_Ratio = *poisson_ratio*] | [Youngs_Modulus = *youngs_modulus*])

Scope

Aria Material

Summary

Pore defect radius from MMP model

Description

The pore defect radius r is computed in two stages. First after decomposition (r_{decomp}) and then after mechanical displacement ($r_{displace}$). After decomposition, the radius is computed as $r_{decomp} = B_o * \phi^{\frac{1}{3}}$ where B_o is the initial half distance between pores and ϕ is the gas volume fraction.

After mechanical displacement, the radius is computed as $r_{displace} = (1 + u_T) r_{decomp}$ where u_T is a dilational displacement field from a thermally induced strain. This displacement field is computed as $u_T = \frac{1-2\nu}{E}(K\varepsilon_T - P_r)$ where ν is Poisson's ratio, E is Young's modulus, K is the bulk modulus, ε_T is a thermally induced strain and P_r is the pressure from chemical decomposition, which is computed from the BKW MMP model for pressure.

Note: You should supply two of the three mechanical terms (ν, E, and K). The third quantity will be calculated using $E = 3K(1-2\nu)$. If you supply all three parameters, they must satisfy this relationship or an error will be thrown.

Lastly, the thermal strain ε_T is computed as $\varepsilon_T = (\beta_0 + \beta_1 T)(T - T_o)$ where T_o is the initial/reference temperature β_0 is the volumetric expansion coefficient and β_1 is a volumetric expansion coefficient multiplier.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
beta0	real	0
beta1	real	0
b_init	real	–
t_init	real	–
BULK_MODUlUS	real	0
poisson_ratio	real	0
youngs_modulus	real	0

Nd_Table

Syntax

Pore Defect Radius [*{of}* *SpeciesName*] = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Pore Defect Radius [*{of}* *SpeciesName*] = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Pore Defect Radius [*{of}* *SpeciesName*] = Scalar_String_Function F = *f*

Scope

Aria Material

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	"string"	–

User_Field

Syntax

Pore Defect Radius [*{of}* *SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Pore Defect Radius [*{of}* *SpeciesName*] = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Scope

Aria Material

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Pore Defect Radius [*{of}* *SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	“string”	–
plugin_parameters	“string”	–

Pore Gas Density

Bkw_Mmp

Syntax

Pore Gas Density [*{of}* *SpeciesName*] = Bkw_Mmp (Initial_Gas_Density = *initial_gas_density* | Rho_C_Init = *rho_c_init* | B_Init = *b_init* | R = *r* | Covol = *covol*)

Scope

Aria Material

Summary

Compute density from MMP model

Description

Computes the density form the MMP model, evaluated as $\rho = \rho_{go}(\frac{r_o}{r})^3 + \rho_{co}(1 - S_f)(\frac{(b_o)^3-(r_o)^3}{(r^3)})$. Here, ρ_{go} is the initial gas density, r_o is the initial pore radius, r is the current defect radius, ρ_{co} is the initial condensed density, S_f is the normalized solid mass fraction and b_0 is the initial half distance between pores. The defect radius r incorporates both chemical decomposition and mechanical distortion

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
initial_gas_density	real	–
rho_c_init	real	–
b_init	real	–
r	real	–
covol	real	–

Constant

Syntax

Pore Gas Density [*{of}* *SpeciesName*] = Constant Value = *value*

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
value	real	–

Copied

Syntax

Pore Gas Density [*{of}* *SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	"string"	–

Exponential

Syntax

Pore Gas Density [*{of}* *SpeciesName*] = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Scope

Aria Material

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Pore Gas Density [*{of}* *SpeciesName*] = Global (Global_Name = *global_name* | [Component = *component*])

Scope

Aria Material

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
global_name	“string”	–
component	integer	0

Nd_Table

Syntax

Pore Gas Density [*{of}* *SpeciesName*] = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Pore Gas Density [*{of}* *SpeciesName*] = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Pore Gas Density [*{of}* *SpeciesName*] = Scalar_String_Function F = *f*

Scope

Aria Material

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	"string"	–

User_Field

Syntax

Pore Gas Density [*{of}* *SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Pore Gas Density [*{of}* *SpeciesName*] = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Scope

Aria Material

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Pore Gas Density [*{of}* *SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	“string”	–
plugin_parameters	“string”	–

Pore Half Distance

Constant

Syntax

Pore Half Distance [*{of}* *SpeciesName*] = Constant Value = *value*

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
value	real	–

Copied

Syntax

Pore Half Distance [*{of}* *SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	“string”	–

Exponential

Syntax

Pore Half Distance [*{of}* *SpeciesName*] = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Scope

Aria Material

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Pore Half Distance [*{of} SpeciesName*] = Global (Global_Name = *global_name* | [Component = *component*])

Scope

Aria Material

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
global_name	"string"	–
component	integer	0

Mmp_Model

Syntax

Pore Half Distance [*{of} SpeciesName*] = Mmp_Model B_Init = *b_init*

Scope

Aria Material

Summary

Half distance between neighboring pores computed by the MMP model.

Description

The half distance between pore defects B computed in as $B = (1 + u_T)r_{decomp}$ where u_T is a displacement field from a thermally induced strain. This displacement field is computed from the MMP model when computing the pore defect radius

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
b_init	real	–

Nd_Table

Syntax

Pore Half Distance [*{of}* *SpeciesName*] = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Pore Half Distance [*{of}* *SpeciesName*] = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Pore Half Distance [*{of}* *SpeciesName*] = Scalar_String_Function F = *f*

Scope

Aria Material

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	"string"	–

User_Field

Syntax

Pore Half Distance [*{of}* *SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Pore Half Distance [*{of}* *SpeciesName*] = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Scope

Aria Material

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Pore Half Distance [*{of} SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	“string”	–
plugin_parameters	“string”	–

Porosity

Constant

Syntax

Porosity [*{of} SpeciesName*] = Constant *{phi | value}* = *phi*

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
phi	real	–

Constant_From_Electrode_Object_Old

Syntax

Porosity [*{of}* *SpeciesName*] = Constant_From_Electrode_Object_Old
Electrodefile = *electrodeFile*

Scope

Aria Material

Summary

This model has been deprecated and removed from the code.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
electrodeFile	"string"	–

Coussy

Syntax

Porosity [*{of}* *SpeciesName*] = Coussy (M_Inv = *m_inv* | Biot = *biot* | [Ref_Porosity = *ref_porosity*] | [Ref_Pressure = *ref_pressure*] | [K_D = *K_d*] | [Thermal_Expansion = *thermal_expansion*] | [Ref_Temperature = *ref_temperature*] | [Alpha_S = *alpha_s*] | [Alpha_F = *alpha_f*] | [K_T = *k_t*])

Scope

Aria Material

Summary

Coussy's model of porosity for deforming media (based on pore pressure and div(u), fixed-stress formulation

Description

$$\phi = \phi_0 + M_{INV}\left(p_{k+1} - p_0\right) + b\epsilon_{v,k} - \alpha_m\left(T_{k+1} - T_0\right) + \frac{b^2}{K_d}\left(p_{k+1} - p_k\right) + b\alpha_s\left(T_{k+1} - T_k\right)$$

where ϕ_0 is the reference porosity, p is pressure, p_0 is reference pressure $\epsilon_{v,k} = \nabla \cdot \mathbf{u}$ ($\mathbf{u}$ is displacement), T is temperature, T_0 is reference temperature, and b is the biot coefficient.

The last two terms are stabilization terms for doing fixed stress iterations (where the mesh motion equation and porosity equations are segregated). In this case, the stabilization parameter K_d must be provided. Additionally, the subscript $k+1$ and k represent the current and old solution, respectively.

When thermal expansion is included, $\alpha_m = \phi_0 \alpha_f + (b - \phi_0)\alpha_s$ and is the bulk thermal expansion coefficient, where α_f and α_s are thermal expansion coefficients for the fluid and solid, respectively.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
m_inv	real	–
biot	real	–
ref_porosity	real	0
ref_pressure	real	0
K_d	real	0
thermal_expansion	integer	0
ref_temperature	real	0
alpha_s	real	0
alpha_f	real	0
k_t	real	0

Deformable_Rock_Compressible

Syntax

Porosity [*{of}* *SpeciesName*] = Deformable_Rock_Compressible (Cr = *cr* | [Ref_Porosity = *ref_porosity*] | [Ref_Pressure = *ref_pressure*] | [Biot = *biot*] | [Clip = *clip*])

Scope

Aria Material

Summary

Rock compressibility linearized model of porosity for deforming media

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
cr	real	–
ref_porosity	real	0
ref_pressure	real	0
biot	real	1
clip	real	1e-06

Deforming

Syntax

Porosity [*{of} SpeciesName*] = Deforming Phi_Init = *phi_init*

Scope

Aria Material

Summary

This model has been deprecated

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
phi_init	real	–

Exponential

Syntax

Porosity [*{of} SpeciesName*] = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Scope

Aria Material

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	“string”	–
constant	real	0
multiplier	real	1
exponent	real	–

From_Density_Ratio

Syntax

Porosity [*{of}* *SpeciesName*] = From_Density_Ratio Solid_Density = *solid_density*

Scope

Aria Material

Summary

Porosity computed as one minus the density ratio between solved for solid-phase density and solid density of non-porous material.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
solid_density	real	–

From_Species

Syntax

Porosity [*{of}* *SpeciesName*] = From_Species ()

Scope

Aria Material

Summary

Computes both the porosity and solid mass fraction. The solid mass fraction is computed using the concentrations of the condensed species, which is then used by the porosity

Description

Computes the normalized solid mass fraction S_f as $S_f = \frac{1}{\rho_b} \sum_{i=1}^{n_{species}} C_i MW_i$. Here, C_i is species i, MW_i is the molecular weight of species i and ρ_b is the bulk density (given by the phase density). The gas volume fraction is then computed as $\phi = (1 - S_f(1 - \phi_o))$, where ϕ_o is the initial gas volume fraction.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–

From_Volume_Fraction_Gas

Syntax

Porosity [*{of}* *SpeciesName*] = From_Volume_Fraction_Gas ()

Scope

Aria Material

Summary

Porosity from the defined volume fraction of gas

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–

Global

Syntax

Porosity [*{of}* *SpeciesName*] = Global (Global_Name = *global_name* | [Component = *component*])

Scope

Aria Material

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
global_name	“string”	–
component	integer	0

Mesh_Deforming

Syntax

Porosity [*{of}* *SpeciesName*] = Mesh_Deforming ([Biot = *biot*] | [Clip = *clip*])

Scope

Aria Material

Summary

Rock compressibility linearized model of porosity for deforming media

Description

Porosity model for a deforming mesh, implemented as

$$\phi = 1 - Bi\left(1 - \frac{\phi_0}{det(F)}\right)$$

Where the Biot parameter (Bi) defaults to 1. An optional clipping tolerance (ϵ) can be provided, so that the result is clipped to lie between ϵ and $1-\epsilon$. If you do not provide a clipping tolerance, Aria will fail the time step if the resulting porosity is outside those bounds with a default ϵ of 10^{-6}.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
biot	real	1
clip	real	1e-06

Modified_Kozeny

Syntax

Porosity [*{of}* *SpeciesName*] = Modified_Kozeny (C = *c* | B1 = *b1* | B2 = *b2* | L_Bed = *L_bed* | R_Bed = *R_bed* | Nx = *nx*)

Scope

Aria Material

Summary

Porosity of a cylindrical particle bed of domain of length L and radius R using experimental model parameters c, $b1$, $b2$ and nx. For 2D $r = x$ and 3D $r = \sqrt{x^2 + z^2}$ with $\bar{r} = r/R$, $\bar{F}1 = 1 - (1 - b1)/2$, $\bar{F}2 = b2 + (1 - b2)/2$, $F1 = 1 - (1 - b1)y/L$, $F2 = b2 + (1 - b2)(1 - \bar{r}^{nx})$, then the porosity $\phi = cF1F2/\bar{F}1\bar{F}2$.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
c	real	–
b1	real	–
b2	real	–
L_bed	real	–
R_bed	real	–
nx	real	–

One_Minus_Volume_Fractions

Syntax

Porosity [*{of}* *SpeciesName*] = One_Minus_Volume_Fractions [Phi_Min = *phi_min*]

Scope

Aria Material

Summary

Porosity computed directly from volume fractions of species.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
phi_min	real	0

Polynomial

Syntax

Porosity [*{of}* *SpeciesName*] = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Rock_Compressible

Syntax

Porosity [*{of}* *SpeciesName*] = Rock_Compressible (Cr = *cr* | [Ref_Porosity = *ref_porosity*] | [Ref_Pressure = *ref_pressure*])

Scope

Aria Material

Summary

Rock compressibility linearized model of porosity

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
cr	real	–
ref_porosity	real	0
ref_pressure	real	0

Scalar_String_Function

Syntax

Porosity [*{of} SpeciesName*] = Scalar_String_Function F = *f*

Scope

Aria Material

Summary

Variable porosity defined by a scalar string function.

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	"string"	–

Solid_Deforming

Syntax

Porosity [*{of} SpeciesName*] = Solid_Deforming ([Biot = *biot*] | [Clip = *clip*])

Scope

Aria Material

Summary

Rock compressibility linearized model of porosity for deforming media

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
biot	real	1
clip	real	1e-06

User_Field

Syntax

Porosity [*{of} SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Porosity [*{of} SpeciesName*] = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Scope

Aria Material

Summary

User Function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

Porosity Def

Constant

Syntax

Porosity Def [*{of} SpeciesName*] = Constant *{phi | value}* = *phi*

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
phi	real	–

Copied

Syntax

Porosity Def [*{of} SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	“string”	–

Deforming

Syntax

Porosity Def [*{of}* *SpeciesName*] = Deforming ()

Scope

Aria Material

Summary

Deforming model for porosity def

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–

Exponential

Syntax

Porosity Def [*{of}* *SpeciesName*] = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Scope

Aria Material

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	“string”	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Porosity Def [*{of} SpeciesName*] = Global (Global_Name = *global_name* | [Component = *component*])

Scope

Aria Material

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
global_name	“string”	–
component	integer	0

Nd_Table

Syntax

Porosity Def [*{of} SpeciesName*] = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Porosity Def [*{of}* *SpeciesName*] = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Porosity Def [*{of}* *SpeciesName*] = Scalar_String_Function F = *f*

Scope

Aria Material

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be

a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	"string"	–

User_Field

Syntax

Porosity Def [*{of}* *SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Porosity Def [*{of}* *SpeciesName*] = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Scope

Aria Material

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1
toggle	"string"	–

User_Plugin

Syntax

Porosity Def [*{of}* *SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	"string"	–
plugin_parameters	"string"	–

Porous Flow System

Air_Water

Syntax

Porous Flow System [*{of}* *SpeciesName*] = Air_Water ([Gx = *gx*] | [Gy = *gy*] | [Gz = *gz*] | [Relative_To = *relative_to*] | [Inverse_Compressibility_Factor = *inverse_compressibility_factor*] | [Water_Gas_Constant = *water_gas_constant*])

Scope

Aria Material

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
gx	real	0
gy	real	0
gz	real	0
relative_to	“string”	–
inverse_compressibility_factor	real	2.17e+06
water_gas_constant	real	461.5

Co2_Brine_Salt

Syntax

Porous Flow System [*{of}* *SpeciesName*] = Co2_Brine_Salt (X_Sat_Co2_File_Name = *x_sat_co2_file_name* | Rho_G_File_Name = *rho_g_file_name* | Rho_Co2_File_Name = *rho_co2_file_name* | [Gx = *gx*] | [Gy = *gy*] | [Gz = *gz*] | [Relative_To = *relative_to*])

Scope

Aria Material

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
x_sat_co2_file_name	“string”	–
rho_g_file_name	“string”	–
rho_co2_file_name	“string”	–
gx	real	0
gy	real	0
gz	real	0
relative_to	“string”	–

Mixed_Single_Phase

Syntax

Porous Flow System [*{of}* *SpeciesName*] = Mixed_Single_Phase ()

Scope

Aria Material

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–

Single_Phase

Syntax

Porous Flow System [*{of}* *SpeciesName*] = Single_Phase ([Gx = *gx*] | [Gy = *gy*] | [Gz = *gz*])

Scope

Aria Material

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
gx	real	0
gy	real	0
gz	real	0

Two_Phase_Immiscible

Syntax

Porous Flow System [*{of}* *SpeciesName*] = Two_Phase_Immiscible ([Gx = *gx*] | [Gy = *gy*] | [Gz = *gz*] | [Relative_To = *relative_to*])

Scope

Aria Material

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
gx	real	0
gy	real	0
gz	real	0
relative_to	“string”	–

Potential

Constant

Syntax

Potential [*{of}* *SpeciesName*] = Constant Value = *value*

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
value	real	–

Copied

Syntax

Potential [*{of}* *SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	“string”	–

Exponential

Syntax

Potential [*{of}* *SpeciesName*] = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Scope

Aria Material

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Potential [*{of} SpeciesName*] = Global (Global_Name = *global_name* | [Component = *component*])

Scope

Aria Material

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Potential [*{of} SpeciesName*] = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Potential [*{of}* *SpeciesName*] = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Potential [*{of} SpeciesName*] = Scalar_String_Function F = *f*

Scope

Aria Material

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	"string"	–

User_Field

Syntax

Potential [*{of}* *SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Potential [*{of}* *SpeciesName*] = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Scope

Aria Material

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Potential [*{of}* *SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	“string”	–
plugin_parameters	“string”	–

Pp

Cvfem_Yplus

Syntax

Pp [*{of}* *SpeciesName*] = Cvfem_Yplus ()

Scope

Aria Material

Summary

CVFEM YPLUS model

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–

Fluid_Traction

Syntax

Pp [*{of}* *SpeciesName*] = Fluid_Traction ()

Scope

Aria Material

Summary

Fluid Traction model

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–

Mesh_Traction

Syntax

Pp [*{of}* *SpeciesName*] = Mesh_Traction ()

Scope

Aria Material

Summary

Mesh Traction model

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–

Mesh_Von_Mises

Syntax

Pp [*{of}* *SpeciesName*] = Mesh_Von_Mises ()

Scope

Aria Material

Summary

Mesh Von Mises model

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–

Solid_Traction

Syntax

Pp [*{of}* *SpeciesName*] = Solid_Traction ()

Scope

Aria Material

Summary

Solid Traction model

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–

Solid_Von_Mises

Syntax

Pp [*{of}* *SpeciesName*] = Solid_Von_Mises ()

Scope

Aria Material

Summary

Solid Von Mises model

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–

Surface_Normal_Shell

Syntax

Pp [*{of}* *SpeciesName*] = Surface_Normal_Shell ()

Scope

Aria Material

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–

Turbulent_Kinetic_Energy_Source

Syntax

Pp [*{of}* *SpeciesName*] = Turbulent_Kinetic_Energy_Source ()

Scope

Aria Material

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–

Vector_Mean_Curvature

Syntax

Pp [*{of}* *SpeciesName*] = Vector_Mean_Curvature ()

Scope

Aria Material

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–

Viscous_Traction

Syntax

Pp [*{of}* *SpeciesName*] = Viscous_Traction ()

Scope

Aria Material

Summary

Viscous Traction model - like Surface_Traction but omits the pressure contribution

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–

Prandtl Number

Constant

Syntax

Prandtl Number [*{of} SpeciesName*] = Constant *{pr | value}* = *Pr*

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Pr	real	–

Copied

Syntax

Prandtl Number [*{of} SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	“string”	–

Correlation

Syntax

Prandtl Number [*{of} SpeciesName*] = Correlation ([Pr = *Pr*] | [Bulk_Node_Interface = *bulk_node_interface*])

Scope

Aria Material

Summary

Correlation Prandtl number computed from material properties

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Pr	real	0
bulk_node_interface	"string"	–

Exponential

Syntax

Prandtl Number [*{of}* *SpeciesName*] = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Scope

Aria Material

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

General

Syntax

Prandtl Number [*{of}* *SpeciesName*] = General ()

Scope

Aria Material

Summary

General Prandtl number computed from material properties

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–

Global

Syntax

Prandtl Number [*{of}* *SpeciesName*] = Global (Global_Name = *global_name* | [Component = *component*])

Scope

Aria Material

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
global_name	“string”	–
component	integer	0

Nd_Table

Syntax

Prandtl Number [*{of}* *SpeciesName*] = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Prandtl Number [*{of}* *SpeciesName*] = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Prandtl Number [*{of}* *SpeciesName*] = Scalar_String_Function F = *f*

Scope

Aria Material

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	"string"	–

User_Field

Syntax

Prandtl Number [*{of}* *SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Prandtl Number [*{of}* *SpeciesName*] = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Scope

Aria Material

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Prandtl Number [*{of} SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	“string”	–
plugin_parameters	“string”	–

Precursor Concentration

Constant

Syntax

Precursor Concentration [*{of} SpeciesName*] = Constant Value = *value*

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
value	real	–

Copied

Syntax

Precursor Concentration [*{of}* *SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	“string”	–

Exponential

Syntax

Precursor Concentration [*{of}* *SpeciesName*] = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Scope

Aria Material

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Precursor Concentration [*{of} SpeciesName*] = Global (Global_Name = *global_name* | [Component = *component*])

Scope

Aria Material

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Precursor Concentration [*{of} SpeciesName*] = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Precursor Concentration [*{of}* *SpeciesName*] = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Precursor Concentration [*{of} SpeciesName*] = Scalar_String_Function F = *f*

Scope

Aria Material

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	"string"	–

User_Field

Syntax

Precursor Concentration [*{of}* *SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Precursor Concentration [*{of}* *SpeciesName*] = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Scope

Aria Material

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Precursor Concentration [*{of} SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	“string”	–
plugin_parameters	“string”	–

Precursor Conservation

Constant

Syntax

Precursor Conservation [*{of} SpeciesName*] = Constant Value = *value*

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
value	real	–

Copied

Syntax

Precursor Conservation [*{of}* *SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	“string”	–

Exponential

Syntax

Precursor Conservation [*{of}* *SpeciesName*] = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Scope

Aria Material

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Precursor Conservation [*{of} SpeciesName*] = Global (Global_Name = *global_name* | [Component = *component*])

Scope

Aria Material

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Precursor Conservation [*{of} SpeciesName*] = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Precursor Conservation [*{of}* *SpeciesName*] = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Precursor Conservation [*{of} SpeciesName*] = Scalar_String_Function F = *f*

Scope

Aria Material

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	"string"	–

User_Field

Syntax

Precursor Conservation [*{of}* *SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Precursor Conservation [*{of}* *SpeciesName*] = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Scope

Aria Material

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Precursor Conservation [*{of}* *SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	“string”	–
plugin_parameters	“string”	–

Pressure

Bkw_Mmp

Syntax

Pressure [*{of}* *SpeciesName*] = Bkw_Mmp (Covol = *covol* | [R = *r*])

Scope

Aria Material

Summary

Computes a spatially varying local pore pressure at nucleation/pore sites. This pressure is computed using the BKW EOS and incorporates chemical decomposition, reaction nucleation and elastic volumetric thermal expansion. Once this local pore pressure exceeds a critical pressure, the chemical material becomes permeable.

Description

Computes the pressure from the BKW EOS, evaluated as $p = \frac{z\rho RT}{MW_g}$. Here, R is the gas constant, T is temperature and MW_g is the molecular weight of the gas species. z is evaluated as $z = (1 + x\exp(0.298x))$, with x computed as $x = \frac{\rho_{pg} 0.0105 V_{co}}{MW_g \sqrt{(T+6620.0)}}$. ρ_{pg} is the PORE GAS DENSITY, and V_{co} is the covolume.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
covol	real	–
r	real	8.314

Constant

Syntax

Pressure [*{of}* *SpeciesName*] = Constant Value = *value*

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
value	real	–

Copied

Syntax

Pressure [*{of}* *SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	“string”	–

Copy_Phase_All

Syntax

Pressure [*{of}* *SpeciesName*] = Copy_Phase_All ()

Scope

Aria Material

Summary

Copy the overall multi-phase pressure to the individual phase.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–

Equation_Of_State

Syntax

Pressure [*{of}* *SpeciesName*] = Equation_Of_State Model_Name = *model_name*

Scope

Aria Material

Summary

Pressure calculated based on species, temperature, and equation of state for the specified pressurization model.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
model_name	“string”	–

Exponential

Syntax

Pressure [*{of}* *SpeciesName*] = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Scope

Aria Material

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

From_Capillary_Pressure

Syntax

Pressure [*{of}* *SpeciesName*] = From_Capillary_Pressure ()

Scope

Aria Material

Summary

Compute the liquid phase pressure = P_gas - P_capillary

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–

From_Ideal_Gas_Density

Syntax

Pressure [*{of}* *SpeciesName*] = From_Ideal_Gas_Density (Mw = *mw* | R = *r*)

Scope

Aria Material

Summary

Calculate pressure using the density and ideal gas law.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
mw	real	–
r	real	–

From_No_Material_Phase

Syntax

Pressure [*{of}* *SpeciesName*] = From_No_Material_Phase ()

Scope

Aria Material

Summary

Copy the overall pressure to the material phase (i.e. gas phase or solid phase).

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–

From_Other_Material_Phase

Syntax

Pressure [*{of}* *SpeciesName*] = From_Other_Material_Phase Other_Phase = *other_phase*

Scope

Aria Material

Summary

Copy the overall pressure from a different phase (i.e. gas phase or solid phase).

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
other_phase	“string”	–

Global

Syntax

Pressure [*{of} SpeciesName*] = Global (Global_Name = *global_name* | [Component = *component*])

Scope

Aria Material

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
global_name	"string"	–
component	integer	0

Ideal_Gas

Syntax

Pressure [*{of} SpeciesName*] = Ideal_Gas R = *r*

Scope

Aria Material

Summary

Calculate the pressure based on the ideal gas law

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
r	real	–

Nd_Table

Syntax

Pressure [*{of} SpeciesName*] = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Pressure [*{of}* *SpeciesName*] = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Pressurization_Model

Syntax

Pressure [*{of}* *SpeciesName*] = Pressurization_Model ()

Scope

Aria Material

Summary

Pressure from a pressurization model

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–

Scalar_String_Function

Syntax

Pressure [*{of}* *SpeciesName*] = Scalar_String_Function F = *f*

Scope

Aria Material

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	"string"	–

User_Field

Syntax

Pressure [*{of}* *SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Pressure [*{of}* *SpeciesName*] = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Scope

Aria Material

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1
toggle	"string"	–

User_Plugin

Syntax

Pressure [*{of} SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	"string"	–
plugin_parameters	"string"	–

Projected Level Set Curvature

Constant

Syntax

Projected Level Set Curvature [*{of} SpeciesName*] = Constant Value = *value*

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
value	real	–

Copied

Syntax

Projected Level Set Curvature [*{of}* *SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	"string"	–

Exponential

Syntax

Projected Level Set Curvature [*{of}* *SpeciesName*] = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Scope

Aria Material

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Projected Level Set Curvature [*{of} SpeciesName*] = Global (Global_Name = *global_name* | [Component = *component*])

Scope

Aria Material

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Projected Level Set Curvature [*{of} SpeciesName*] = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Projected Level Set Curvature [*{of} SpeciesName*] = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Projected Level Set Curvature [*{of}* *SpeciesName*] = Scalar_String_Function F = *f*

Scope

Aria Material

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	"string"	–

User_Field

Syntax

Projected Level Set Curvature [*{of}* *SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Projected Level Set Curvature [*{of}* *SpeciesName*] = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Scope

Aria Material

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Projected Level Set Curvature [*{of} SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	“string”	–
plugin_parameters	“string”	–

Projected Pressure Gradient

Constant

Syntax

Projected Pressure Gradient [*{of} SpeciesName*] = Constant ([Valuex = *valueX*] | [Valuey = *valueY*] | [Valuez = *valueZ*])

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
valueX	real	0
valueY	real	0
valueZ	real	0

Copied

Syntax

Projected Pressure Gradient [*{of}* *SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	“string”	–

Scalar_Vector_Product

Syntax

Projected Pressure Gradient [*{of}* *SpeciesName*] = Scalar_Vector_Product (Vector = *vector* | [Scalars = *scalars*] | [Multiplier = *multiplier*])

Scope

Aria Material

Summary

Scalar vector product

Description

Multiplies a vector by a list of provided scalars (variables) and a multiplier (constant)

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
vector	"string"	–
scalars	"string"	–
multiplier	real	1

Tensor_Vector_Product

Syntax

Projected Pressure Gradient [*{of}* *SpeciesName*] = Tensor_Vector_Product (Vector = *vector* | Tensor = *tensor* | [Scale = *scale*])

Scope

Aria Material

Summary

Tensor vector product

Description

Calculates a vector resulting from a tensor-vector product with an optional scaling factor.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
vector	"string"	–
tensor	"string"	–
scale	real	1

User_Field

Syntax

Projected Pressure Gradient [*{of}* *SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Vector from User Field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Projected Pressure Gradient [*{of} SpeciesName*] = User_Function ([Name_X = *name_x*] | [Name_Y = *name_y*] | [Name_Z = *name_z*] | X = *x* | [Multiplier = *multiplier*])

Scope

Aria Material

Summary

Value from a Vector User Function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name_x	"string"	–
name_y	"string"	–
name_z	"string"	–
x	"string"	–
multiplier	real	1

User_Plugin

Syntax

Projected Pressure Gradient [*{of} SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	“string”	–
plugin_parameters	“string”	–

Vector_String_Function

Syntax

Projected Pressure Gradient [*{of}* *SpeciesName*] = Vector_String_Function (F_X = *f_x* | F_Y = *f_y* | [F_Z = *f_z*])

Scope

Aria Material

Summary

Value from a string function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f_x	“string”	–
f_y	“string”	–
f_z	“string”	–

Projected Pressure Stab Tau

Constant

Syntax

Projected Pressure Stab Tau [*{of}* *SpeciesName*] = Constant Value = *value*

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
value	real	–

Copied

Syntax

Projected Pressure Stab Tau [*{of} SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	"string"	–

Exponential

Syntax

Projected Pressure Stab Tau [*{of} SpeciesName*] = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Scope

Aria Material

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Projected Pressure Stab Tau [*{of} SpeciesName*] = Global (Global_Name = *global_name* | [Component = *component*])

Scope

Aria Material

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
global_name	“string”	–
component	integer	0

Nd_Table

Syntax

Projected Pressure Stab Tau [*{of} SpeciesName*] = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Projected Pressure Stab Tau [*{of}* *SpeciesName*] = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Projected Pressure Stab Tau [*{of}* *SpeciesName*] = Scalar_String_Function F = *f*

Scope

Aria Material

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	"string"	–

User_Field

Syntax

Projected Pressure Stab Tau [*{of} SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Projected Pressure Stab Tau [*{of} SpeciesName*] = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Scope

Aria Material

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1
toggle	"string"	–

User_Plugin

Syntax

Projected Pressure Stab Tau [*{of} SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	"string"	–
plugin_parameters	"string"	–

Projected Velocity

Constant

Syntax

Projected Velocity [*{of} SpeciesName*] = Constant ([X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
x	real	0
y	real	0
z	real	0

Copied

Syntax

Projected Velocity [*{of}* *SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	“string”	–

User_Field

Syntax

Projected Velocity [*{of}* *SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Vector from User Field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Projected Velocity [*{of}* *SpeciesName*] = User_Function ([Name_X = *name_x*] | [Name_Y = *name_y*] | [Name_Z = *name_z*] | X = *x* | [Multiplier = *multiplier*])

Scope

Aria Material

Summary

Value from a Vector User Function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name_x	"string"	–
name_y	"string"	–
name_z	"string"	–
x	"string"	–
multiplier	real	1

User_Plugin

Syntax

Projected Velocity [*{of}* *SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	"string"	–
plugin_parameters	"string"	–

Vector_String_Function

Syntax

Projected Velocity [*{of}* *SpeciesName*] = Vector_String_Function (F_X = *f_x* | F_Y = *f_y* | [F_Z = *f_z*])

Scope

Aria Material

Summary

Value from a string function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f_x	"string"	–
f_y	"string"	–
f_z	"string"	–

Projected Velocity X

Constant

Syntax

Projected Velocity X [*{of}* *SpeciesName*] = Constant Value = *value*

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
value	real	–

Copied

Syntax

Projected Velocity X [*{of}* *SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	"string"	–

Exponential

Syntax

Projected Velocity X [*{of}* *SpeciesName*] = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Scope

Aria Material

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Projected Velocity X [*{of}* *SpeciesName*] = Global (Global_Name = *global_name* | [Component = *component*])

Scope

Aria Material

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
global_name	“string”	–
component	integer	0

Nd_Table

Syntax

Projected Velocity X [*{of}* *SpeciesName*] = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Projected Velocity X [*{of}* *SpeciesName*] = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Projected Velocity X [*{of}* *SpeciesName*] = Scalar_String_Function F = *f*

Scope

Aria Material

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be

a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	"string"	–

User_Field

Syntax

Projected Velocity X [*{of}* *SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Projected Velocity X [*{of}* *SpeciesName*] = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Scope

Aria Material

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1
toggle	"string"	–

User_Plugin

Syntax

Projected Velocity X [*{of} SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	"string"	–
plugin_parameters	"string"	–

Projected Velocity Y

Constant

Syntax

Projected Velocity Y [*{of} SpeciesName*] = Constant Value = *value*

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
value	real	–

Copied

Syntax

Projected Velocity Y [*{of}* *SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	“string”	–

Exponential

Syntax

Projected Velocity Y [*{of}* *SpeciesName*] = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Scope

Aria Material

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Projected Velocity Y [*{of} SpeciesName*] = Global (Global_Name = *global_name* | [Component = *component*])

Scope

Aria Material

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Projected Velocity Y [*{of} SpeciesName*] = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Projected Velocity Y [*{of}* *SpeciesName*] = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Projected Velocity Y [*{of}* *SpeciesName*] = Scalar_String_Function F = *f*

Scope

Aria Material

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	"string"	–

User_Field

Syntax

Projected Velocity Y [*{of}* *SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Projected Velocity Y [*{of}* *SpeciesName*] = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Scope

Aria Material

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Projected Velocity Y [*{of} SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	“string”	–
plugin_parameters	“string”	–

Projected Velocity Z

Constant

Syntax

Projected Velocity Z [*{of} SpeciesName*] = Constant Value = *value*

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
value	real	–

Copied

Syntax

Projected Velocity Z [*{of}* *SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	“string”	–

Exponential

Syntax

Projected Velocity Z [*{of}* *SpeciesName*] = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Scope

Aria Material

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Projected Velocity Z [*{of} SpeciesName*] = Global (Global_Name = *global_name* | [Component = *component*])

Scope

Aria Material

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Projected Velocity Z [*{of} SpeciesName*] = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Projected Velocity Z [*{of}* *SpeciesName*] = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Projected Velocity Z [*{of}* *SpeciesName*] = Scalar_String_Function F = *f*

Scope

Aria Material

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	"string"	–

User_Field

Syntax

Projected Velocity Z [*{of}* *SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Projected Velocity Z [*{of}* *SpeciesName*] = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Scope

Aria Material

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Projected Velocity Z [*{of} SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	“string”	–
plugin_parameters	“string”	–

R

Radiation Form Factor

Calore_User_Sub

Syntax

Radiation Form Factor [*{of} SpeciesName*] = Calore_User_Sub (Name = *name* | Type = *type* | [Multiplier = *multiplier*] | [Material_Data_Block = *material_data_block*] | [Data = *data*] | [Scaling_Field = *scaling_field*])

Scope

Aria Material

Summary

Values from a Calore user subroutine

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
type	“string”	–
multiplier	real	1
material_data_block	“string”	–
data	“string”	–
scaling_field	“string”	–

Constant

Syntax

Radiation Form Factor [*{of} SpeciesName*] = Constant *{f | value}* = *f*

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	real	–

Copied

Syntax

Radiation Form Factor [*{of} SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	“string”	–

Exponential

Syntax

Radiation Form Factor [*{of}* *SpeciesName*] = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Scope

Aria Material

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Fortran

Syntax

Radiation Form Factor [*{of}* *SpeciesName*] = Fortran ([Multiplier = *multiplier*] | Sub_Name = *sub_name* | [Real_Data = *real_data*] | [Int_Data = *int_data*] | [Resource_Name = *resource_name*] | [Data = *data*])

Scope

Aria Material

Summary

Fortran subroutine emissivity.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
multiplier	real	1
sub_name	“string”	–
real_data	“string”	–
int_data	“string”	–
resource_name	“string”	–
data	“string”	–

Global

Syntax

Radiation Form Factor [*{of}* *SpeciesName*] = Global (Global_Name = *global_name* | [Component = *component*])

Scope

Aria Material

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
global_name	“string”	–
component	integer	0

Nd_Table

Syntax

Radiation Form Factor [*{of}* *SpeciesName*] = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Radiation Form Factor [*{of}* *SpeciesName*] = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Radiation Form Factor [*{of}* *SpeciesName*] = Scalar_String_Function F = *f*

Scope

Aria Material

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	"string"	–

User_Field

Syntax

Radiation Form Factor [*{of} SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Radiation Form Factor [*{of} SpeciesName*] = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Scope

Aria Material

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Radiation Form Factor [*{of}* *SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	“string”	–
plugin_parameters	“string”	–

Radiation Source

Black_Body

Syntax

Radiation Source [*{of}* *SpeciesName*] = Black_Body ()

Scope

Aria Material

Summary

Source due to black body radiation

Description

$$S = a\frac{\sigma T^4}{\pi}$$

a is the absorption cross section expression, sigma is the Stefan-Boltzmann constant, and T is the temperature expression.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–

Constant

Syntax

Radiation Source [*{of}* *SpeciesName*] = Constant *{s | value}* = *s*

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
s	real	–

Copied

Syntax

Radiation Source [*{of}* *SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	“string”	–

Exponential

Syntax

Radiation Source [*{of} SpeciesName*] = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Scope

Aria Material

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Radiation Source [*{of} SpeciesName*] = Global (Global_Name = *global_name* | [Component = *component*])

Scope

Aria Material

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Radiation Source [*{of} SpeciesName*] = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Radiation Source [*{of} SpeciesName*] = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Radiation Source [*{of}* *SpeciesName*] = Scalar_String_Function F = *f*

Scope

Aria Material

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	"string"	–

User_Field

Syntax

Radiation Source [*{of}* *SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Radiation Source [*{of}* *SpeciesName*] = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Scope

Aria Material

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Radiation Source [*{of}* *SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	“string”	–
plugin_parameters	“string”	–

Radiative Conductivity

Chemeq_Foam

Syntax

Radiative Conductivity [*{of}* *SpeciesName*] = Chemeq_Foam (Kc0 = *Kc0* | Kc1 = *Kc1*)

Scope

Aria Material

Summary

Radiation conductivity for ChemEq foams, adjusting for the mass fraction of solids in the foam

Description

$$\kappa_{rad} = \sigma \left(\frac{K_{c,0}}{\Sigma \text{convert}(c_{\text{ChemEq}}) + K_{c,1}} \right) T^3$$

where σ is the Stefan-Boltzmann constant, the convert() function evaluates the appropriate conversion from the ChemEq model concentration units to mass fraction, and the summation is over the solid phase species in the ChemEq model.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Kc0	real	–
Kc1	real	–

Constant

Syntax

Radiative Conductivity [*{of} SpeciesName*] = Constant *{krad | value}* = *Krad*

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Krad	real	–

Copied

Syntax

Radiative Conductivity [*{of} SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	"string"	–

Exponential

Syntax

Radiative Conductivity [*{of}* *SpeciesName*] = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Scope

Aria Material

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Radiative Conductivity [*{of}* *SpeciesName*] = Global (Global_Name = *global_name* | [Component = *component*])

Scope

Aria Material

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
global_name	“string”	–
component	integer	0

Nd_Table

Syntax

Radiative Conductivity [*{of}* *SpeciesName*] = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Optically_Thick

Syntax

Radiative Conductivity [*{of}* *SpeciesName*] = Optically_Thick ([Beta_R = *beta_r*] | [N = *n*])

Scope

Aria Material

Summary

Radiation conductivity for optically thick materials

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
beta_r	real	1
n	real	1

Polynomial

Syntax

Radiative Conductivity [*{of}* *SpeciesName*] = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Radiative Conductivity [*{of}* *SpeciesName*] = Scalar_String_Function F = *f*

Scope

Aria Material

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	"string"	–

User_Field

Syntax

Radiative Conductivity [*{of}* *SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Radiative Conductivity [*{of}* *SpeciesName*] = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Scope

Aria Material

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1
toggle	"string"	–

User_Plugin

Syntax

Radiative Conductivity [*{of}* *SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	"string"	–
plugin_parameters	"string"	–

Volume_Average

Syntax

Radiative Conductivity [*{of}* *SpeciesName*] = Volume_Average ()

Scope

Aria Material

Summary

Radiation conductivity computed from normalized volume average of species radiation conductivities

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–

Reaction Rate Multiplier

Constant

Syntax

Reaction Rate Multiplier [*{of}* *SpeciesName*] = Constant Value = *value*

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
value	real	–

Copied

Syntax

Reaction Rate Multiplier [*{of}* *SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	“string”	–

Electrode_Object

Syntax

Reaction Rate Multiplier [*{of}* *SpeciesName*] = Electrode_Object [Smin = *Smin*]

Scope

Aria Material

Summary

Rate multiplier based on melting of the electrolyte and global deactivation of the electrodes

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Smin	real	0.01

Exponential

Syntax

Reaction Rate Multiplier [*{of}* *SpeciesName*] = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Scope

Aria Material

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Reaction Rate Multiplier [*{of}* *SpeciesName*] = Global (Global_Name = *global_name* | [Component = *component*])

Scope

Aria Material

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Reaction Rate Multiplier [*{of}* *SpeciesName*] = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

One_Minus_Gas_Volume_Fraction

Syntax

Reaction Rate Multiplier [*{of}* *SpeciesName*] =
One_Minus_Gas_Volume_Fraction ()

Scope

Aria Material

Summary

Compute a multiplier that is one minus the gas volume fractions

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–

Polynomial

Syntax

Reaction Rate Multiplier [*{of}* *SpeciesName*] = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Reaction Rate Multiplier [*{of}* *SpeciesName*] = Scalar_String_Function F = *f*

Scope

Aria Material

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	"string"	–

User_Field

Syntax

Reaction Rate Multiplier [*{of} SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Reaction Rate Multiplier [*{of} SpeciesName*] = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Scope

Aria Material

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Reaction Rate Multiplier [*{of}* *SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	“string”	–
plugin_parameters	“string”	–

Vitrification

Syntax

Reaction Rate Multiplier [*{of}* *SpeciesName*] = Vitrification (W = *w* | Beta = *beta*)

Scope

Aria Material

Summary

Compute vitrification rate multiplier from polymerization shift factor

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
w	real	–
beta	real	–

Refractive Index Ratio

Constant

Syntax

Refractive Index Ratio [*{of}* *SpeciesName*] = Constant *{value | varepsilon}* = *varepsilon*

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
varepsilon	real	–

Copied

Syntax

Refractive Index Ratio [*{of}* *SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	“string”	–

Exponential

Syntax

Refractive Index Ratio [*{of}* *SpeciesName*] = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Scope

Aria Material

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Refractive Index Ratio [*{of}* *SpeciesName*] = Global (Global_Name = *global_name* | [Component = *component*])

Scope

Aria Material

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Refractive Index Ratio [*{of}* *SpeciesName*] = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Refractive Index Ratio [*{of}* *SpeciesName*] = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Refractive Index Ratio [*{of}* *SpeciesName*] = Scalar_String_Function F = *f*

Scope

Aria Material

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	"string"	–

User_Field

Syntax

Refractive Index Ratio [*{of}* *SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Refractive Index Ratio [*{of}* *SpeciesName*] = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Scope

Aria Material

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Refractive Index Ratio [*{of}* *SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	“string”	–
plugin_parameters	“string”	–

Relative Permeability

Brooks_Corey

Syntax

Relative Permeability [*{of}* *SpeciesName*] = Brooks_Corey Liquid_Phase_Name = *liquid_phase_name*

Scope

Aria Material

Summary

A Brooks-Corey liquid/gas model for relative permeability

Description

Residual saturations for liquid and gas are S_{lr} and S_{gr}.

Hence the maximum possible liquid saturation is $S_{lm} = 1 - S_{gr}$ and the effective saturation is $S_e = (S_l - S_{lr})/(S_{lm} - S_{lr})$ where S_l is the wetting phase saturation. The pore size distribution index for the media is λ.

For wetting phase, if $S_l >= S_{lm}$, then $K_l = 1$, else if $S_l <= S_{lr}$ then $K_l = 0$, else $K_l = S_e^{3+2/\lambda}$.

For nonwetting phase, if $S_l >= S_{lm}$, then $K_g = 0$, else if $S_l <= S_{lr}$ then $K_g = 1$, else $K_g = (1 - S_e)^2(1 - S_e^{1+2/\lambda})$.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
liquid_phase_name	"string"	–

Constant

Syntax

Relative Permeability [*{of}* *SpeciesName*] = Constant *{k | value}* = *k*

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
k	real	–

Copied

Syntax

Relative Permeability [*{of}* *SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	"string"	–

Exponential

Syntax

Relative Permeability [*{of}* *SpeciesName*] = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Scope

Aria Material

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Relative Permeability [*{of}* *SpeciesName*] = Global (Global_Name = *global_name* | [Component = *component*])

Scope

Aria Material

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
global_name	“string”	–
component	integer	0

Mass_Average

Syntax

Relative Permeability [*{of}* *SpeciesName*] = Mass_Average ()

Scope

Aria Material

Summary

Linear mixture-averaged mass diffusivity

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–

Nd_Table

Syntax

Relative Permeability [*{of}* *SpeciesName*] = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Relative Permeability [*{of}* *SpeciesName*] = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Relative Permeability [*{of}* *SpeciesName*] = Scalar_String_Function F = *f*

Scope

Aria Material

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	"string"	–

T_Exponent

Syntax

Relative Permeability [*{of}* *SpeciesName*] = T_Exponent (K_Ref = *k_ref* | T_Ref = *t_ref* | N = *n*)

Scope

Aria Material

Summary

Relative permeability as a power of normalized temperature

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
k_ref	real	–
t_ref	real	–
n	real	–

Udell_Cubic_Gas

Syntax

Relative Permeability [*{of}* *SpeciesName*] = Udell_Cubic_Gas (Liquid_Phase_Name = *liquid_phase_name* | Residual_Liquid_Phase_Saturation = *residual_liquid_phase_saturation* | Residual_Gas_Phase_Saturation = *residual_gas_phase_saturation*)

Scope

Aria Material

Summary

A Udell cubic gas model for relative permeability

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
liquid_phase_name	“string”	–
residual_liquid_phase_saturation	real	–
residual_gas_phase_saturation	real	–

Udell_Cubic_Liquid

Syntax

Relative Permeability [*{of}* *SpeciesName*] = Udell_Cubic_Liquid
(Liquid_Phase_Name = *liquid_phase_name* |
Residual_Liquid_Phase_Saturation = *residual_liquid_phase_saturation* |
Residual_Gas_Phase_Saturation = *residual_gas_phase_saturation*)

Scope

Aria Material

Summary

A Udell cubic liquid model for relative permeability

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
liquid_phase_name	"string"	–
residual_liquid_phase_saturation	real	–
residual_gas_phase_saturation	real	–

User_Field

Syntax

Relative Permeability [*{of}* *SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Relative Permeability [*{of}* *SpeciesName*] = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Scope

Aria Material

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1
toggle	"string"	–

User_Plugin

Syntax

Relative Permeability [*{of}* *SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	"string"	–
plugin_parameters	"string"	–

Van_Genuchten_Gas

Syntax

Relative Permeability [*{of}* *SpeciesName*] = Van_Genuchten_Gas (Liquid_Phase_Name = *liquid_phase_name* | [Beta_Field = *beta_field*] | [Sgr_Field = *sgr_field*] | [Slr_Field = *slr_field*] | [Beta = *beta*] | [Residual_Liquid_Phase_Saturation = *residual_liquid_phase_saturation*] | [Residual_Gas_Phase_Saturation = *residual_gas_phase_saturation*])

Scope

Aria Material

Summary

A Van Genuchten gas model for relative permeability

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
liquid_phase_name	"string"	–
beta_field	"string"	–
sgr_field	"string"	–
slr_field	"string"	–
beta	real	0
residual_liquid_phase_saturation	real	0
residual_gas_phase_saturation	real	0

Van_Genuchten_Liquid

Syntax

Relative Permeability [*{of}* *SpeciesName*] = Van_Genuchten_Liquid (Liquid_Phase_Name = *liquid_phase_name* | [Beta_Field = *beta_field*] | [Sgr_Field = *sgr_field*] | [Slr_Field = *slr_field*] | [Beta = *beta*] | [Residual_Liquid_Phase_Saturation = *residual_liquid_phase_saturation*] | [Residual_Gas_Phase_Saturation = *residual_gas_phase_saturation*])

Scope

Aria Material

Summary

A Van Genuchten liquid model for relative permeability

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
liquid_phase_name	"string"	–
beta_field	"string"	–
sgr_field	"string"	–
slr_field	"string"	–
beta	real	0
residual_liquid_phase_saturation	real	0
residual_gas_phase_saturation	real	0

Volume_Average

Syntax

Relative Permeability [*{of} SpeciesName*] = Volume_Average ()

Scope

Aria Material

Summary

Relative permeability computed from volume average of species relative permeabilities

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–

Reservoir Depth

Constant

Syntax

Reservoir Depth [*{of} SpeciesName*] = Constant *{h | value}* = *h*

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
h	real	–

Copied

Syntax

Reservoir Depth [*{of}* *SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	“string”	–

Exponential

Syntax

Reservoir Depth [*{of}* *SpeciesName*] = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Scope

Aria Material

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Reservoir Depth [*{of}* *SpeciesName*] = Global (Global_Name = *global_name* | [Component = *component*])

Scope

Aria Material

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Reservoir Depth [*{of}* *SpeciesName*] = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Reservoir Depth [*{of}* *SpeciesName*] = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Reservoir Depth [*{of}* *SpeciesName*] = Scalar_String_Function F = *f*

Scope

Aria Material

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	"string"	–

User_Field

Syntax

Reservoir Depth [*{of}* *SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Reservoir Depth [*{of}* *SpeciesName*] = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Scope

Aria Material

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Reservoir Depth [*{of}* *SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	“string”	–
plugin_parameters	“string”	–

Residual Saturation

Constant

Syntax

Residual Saturation [*{of}* *SpeciesName*] = Constant Value = *value*

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
value	real	–

Copied

Syntax

Residual Saturation [*{of}* *SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	“string”	–

Exponential

Syntax

Residual Saturation [*{of}* *SpeciesName*] = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Scope

Aria Material

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Residual Saturation [*{of} SpeciesName*] = Global (Global_Name = *global_name* | [Component = *component*])

Scope

Aria Material

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Residual Saturation [*{of} SpeciesName*] = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Residual Saturation [*{of}* *SpeciesName*] = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Residual Saturation [*{of}* *SpeciesName*] = Scalar_String_Function F = *f*

Scope

Aria Material

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	"string"	–

User_Field

Syntax

Residual Saturation [*{of}* *SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Residual Saturation [*{of}* *SpeciesName*] = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Scope

Aria Material

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Residual Saturation [*{of} SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	“string”	–
plugin_parameters	“string”	–

Retardation

Constant

Syntax

Retardation [*{of} SpeciesName*] = Constant *{r | value}* = *r*

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
r	real	–

Copied

Syntax

Retardation [*{of}* *SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	“string”	–

Exponential

Syntax

Retardation [*{of}* *SpeciesName*] = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Scope

Aria Material

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Gas_Phase_Distributed

Syntax

Retardation [*{of}* *SpeciesName*] = Gas_Phase_Distributed ()

Scope

Aria Material

Summary

Distributed retardation factor for gas phase in porous flow. The corresponding distribution coefficient should be defined with a DISTRIBUTION COEFFICIENT model.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–

Global

Syntax

Retardation [*{of}* *SpeciesName*] = Global (Global_Name = *global_name* | [Component = *component*])

Scope

Aria Material

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
global_name	"string"	–
component	integer	0

Liquid_Phase_Distributed

Syntax

Retardation [*{of} SpeciesName*] = Liquid_Phase_Distributed ()

Scope

Aria Material

Summary

Distributed retardation factor for liquid phase porous flow. The corresponding distribution coefficient should be defined with a DISTRIBUTION COEFFICIENT model.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–

Nd_Table

Syntax

Retardation [*{of} SpeciesName*] = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Nonwetting_Phase_Distributed

Syntax

Retardation [*{of}* *SpeciesName*] = Nonwetting_Phase_Distributed ()

Scope

Aria Material

Summary

Distributed retardation factor for non-wetting phase in porous flow. The corresponding distribution coefficient should be defined with a DISTRIBUTION COEFFICIENT model.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–

Polynomial

Syntax

Retardation [*{of}* *SpeciesName*] = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Saturated_Distributed

Syntax

Retardation [*{of}* *SpeciesName*] = Saturated_Distributed ()

Scope

Aria Material

Summary

This model has been deprecated and removed from the code.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–

Scalar_String_Function

Syntax

Retardation [*{of}* *SpeciesName*] = Scalar_String_Function F = *f*

Scope

Aria Material

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	"string"	–

User_Field

Syntax

Retardation [*{of}* *SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Retardation [*{of}* *SpeciesName*] = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Scope

Aria Material

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1
toggle	"string"	–

User_Plugin

Syntax

Retardation [*{of} SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	"string"	–
plugin_parameters	"string"	–

Wetting_Phase_Distributed

Syntax

Retardation [*{of} SpeciesName*] = Wetting_Phase_Distributed ()

Scope

Aria Material

Summary

Distributed retardation factor for wetting phase porous flow. The corresponding distribution coefficient should be defined with a DISTRIBUTION COEFFICIENT model.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–

Reynolds Number

Constant

Syntax

Reynolds Number [*{of}* *SpeciesName*] = Constant *{re | value}* = *Re*

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Re	real	–

Copied

Syntax

Reynolds Number [*{of}* *SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	“string”	–

Correlation

Syntax

Reynolds Number [*{of}* *SpeciesName*] = Correlation ([Re = *Re*] | [Bulk_Node_Interface = *bulk_node_interface*])

Scope

Aria Material

Summary

Correlation Reynolds number computed from material properties

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Re	real	0
bulk_node_interface	"string"	–

Exponential

Syntax

Reynolds Number [*{of}* *SpeciesName*] = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Scope

Aria Material

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

General

Syntax

Reynolds Number [*{of}* *SpeciesName*] = General L = *l*

Scope

Aria Material

Summary

Reynold number computed from a specified length scale

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
l	real	–

Global

Syntax

Reynolds Number [*{of}* *SpeciesName*] = Global (Global_Name = *global_name* | [Component = *component*])

Scope

Aria Material

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
global_name	“string”	–
component	integer	0

Nd_Table

Syntax

Reynolds Number [*{of}* *SpeciesName*] = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Reynolds Number [*{of} SpeciesName*] = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Reynolds Number [*{of}* *SpeciesName*] = Scalar_String_Function F = *f*

Scope

Aria Material

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	"string"	–

User_Field

Syntax

Reynolds Number [*{of}* *SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Reynolds Number [*{of}* *SpeciesName*] = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Scope

Aria Material

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Reynolds Number [*{of} SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	“string”	–
plugin_parameters	“string”	–

S

Saturated Retardation

Constant

Syntax

Saturated Retardation [*{of} SpeciesName*] = Constant *{r | value}* = *r*

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
r	real	–

Copied

Syntax

Saturated Retardation [*{of}* *SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	“string”	–

Exponential

Syntax

Saturated Retardation [*{of}* *SpeciesName*] = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Scope

Aria Material

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Saturated Retardation [*{of} SpeciesName*] = Global (Global_Name = *global_name* | [Component = *component*])

Scope

Aria Material

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Saturated Retardation [*{of} SpeciesName*] = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Saturated Retardation [*{of}* *SpeciesName*] = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Saturated Retardation [*{of} SpeciesName*] = Scalar_String_Function F = *f*

Scope

Aria Material

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	"string"	–

User_Field

Syntax

Saturated Retardation [*{of} SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Saturated Retardation [*{of} SpeciesName*] = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Scope

Aria Material

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Saturated Retardation [*{of} SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	“string”	–
plugin_parameters	“string”	–

Saturation

Constant

Syntax

Saturation [*{of} SpeciesName*] = Constant *{s | value}* = *s*

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
s	real	–

Copied

Syntax

Saturation [*{of}* *SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	“string”	–

Exponential

Syntax

Saturation [*{of}* *SpeciesName*] = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Scope

Aria Material

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

From_Bulk_Species

Syntax

Saturation [*{of}* *SpeciesName*] = From_Bulk_Species ([Min = *min*] | [Max = *max*])

Scope

Aria Material

Summary

Saturation computed from the bulk species concentrations

Description

Calculates the saturation from the bulk species concentrations ($c_{b,i}$), partial molar volumes (v_i), and porosity (ϕ).

$$S = \frac{1}{\phi} \sum_i c_{b,i} v_i$$

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
min	real	0
max	real	1

From_Other_Phase

Syntax

Saturation [*{of}* *SpeciesName*] = From_Other_Phase ()

Scope

Aria Material

Summary

Get the saturation from one minus the other phase saturation

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–

Global

Syntax

Saturation [*{of}* *SpeciesName*] = Global (Global_Name = *global_name* | [Component = *component*])

Scope

Aria Material

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
global_name	“string”	–
component	integer	0

Nd_Table

Syntax

Saturation [*{of}* *SpeciesName*] = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Saturation [*{of}* *SpeciesName*] = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Saturation [*{of} SpeciesName*] = Scalar_String_Function F = *f*

Scope

Aria Material

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	"string"	–

User_Field

Syntax

Saturation [*{of}* *SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Saturation [*{of}* *SpeciesName*] = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Scope

Aria Material

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Saturation [*{of}* *SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	“string”	–
plugin_parameters	“string”	–

Van_Genuchten

Syntax

Saturation [*{of}* *SpeciesName*] = Van_Genuchten ([Is_Wetting_Phase = *is_wetting_phase*] | [Nonwetting_Phase = *nonwetting_phase*] | Reference_Capillary_Pressure = *reference_capillary_pressure* | Beta = *beta* | Residual_Wetting_Phase_Saturation = *residual_wetting_phase_saturation* | Residual_Nonwetting_Phase_Saturation = *residual_nonwetting_phase_saturation* | [Pcstar = *PcStar*] | [Pcmax = *PcMax*])

Scope

Aria Material

Summary

A Van Genuchten model for saturation using capillary pressure

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
is_wetting_phase	integer	0
nonwetting_phase	“string”	NONWETTING_PHASE
reference_capillary_pressure	real	–
beta	real	–
residual_wetting_phase_saturation	real	–
residual_nonwetting_phase_saturation	real	–
PcStar	real	1e+06
PcMax	real	1e+06

Scaling

Constant

Syntax

Scaling [*{of} SpeciesName*] = Constant Value = *value*

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
value	real	–

Copied

Syntax

Scaling [*{of} SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	"string"	–

Exponential

Syntax

Scaling [*{of}* *SpeciesName*] = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Scope

Aria Material

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Scaling [*{of}* *SpeciesName*] = Global (Global_Name = *global_name* | [Component = *component*])

Scope

Aria Material

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
global_name	“string”	–
component	integer	0

Nd_Table

Syntax

Scaling [*{of}* *SpeciesName*] = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Scaling [*{of}* *SpeciesName*] = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Scaling [*{of}* *SpeciesName*] = Scalar_String_Function F = *f*

Scope

Aria Material

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	"string"	–

User_Field

Syntax

Scaling [*{of}* *SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Scaling [*{of}* *SpeciesName*] = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Scope

Aria Material

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Scaling [*{of}* *SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	“string”	–
plugin_parameters	“string”	–

Scattering Coefficient

Constant

Syntax

Scattering Coefficient [*{of}* *SpeciesName*] = Constant Value = *value*

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
value	real	–

Copied

Syntax

Scattering Coefficient [*{of}* *SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	“string”	–

Exponential

Syntax

Scattering Coefficient [*{of}* *SpeciesName*] = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Scope

Aria Material

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Scattering Coefficient [*{of} SpeciesName*] = Global (Global_Name = *global_name* | [Component = *component*])

Scope

Aria Material

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Scattering Coefficient [*{of} SpeciesName*] = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Scattering Coefficient [*{of}* *SpeciesName*] = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Scattering Coefficient [*{of} SpeciesName*] = Scalar_String_Function F = *f*

Scope

Aria Material

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	"string"	–

User_Field

Syntax

Scattering Coefficient [*{of}* *SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Scattering Coefficient [*{of}* *SpeciesName*] = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Scope

Aria Material

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Scattering Coefficient [*{of} SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	“string”	–
plugin_parameters	“string”	–

Scattering Cross Section

Calore_User_Sub

Syntax

Scattering Cross Section [*{of} SpeciesName*] = Calore_User_Sub (Name = *name* | Type = *type* | [Multiplier = *multiplier*] | [Material_Data_Block = *material_data_block*] | [Data = *data*] | [Scaling_Field = *scaling_field*])

Scope

Aria Material

Summary

Calore user subroutine scattering cross section

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
type	“string”	–
multiplier	real	1
material_data_block	“string”	–
data	“string”	–
scaling_field	“string”	–

Constant

Syntax

Scattering Cross Section [*{of}* *SpeciesName*] = Constant *{scat | value}* = *scat*

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
scat	real	–

Copied

Syntax

Scattering Cross Section [*{of}* *SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	“string”	–

Exponential

Syntax

Scattering Cross Section [*{of}* *SpeciesName*] = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Scope

Aria Material

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Scattering Cross Section [*{of}* *SpeciesName*] = Global (Global_Name = *global_name* | [Component = *component*])

Scope

Aria Material

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Scattering Cross Section [*{of}* *SpeciesName*] = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Scattering Cross Section [*{of}* *SpeciesName*] = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Scattering Cross Section [*{of}* *SpeciesName*] = Scalar_String_Function F = *f*

Scope

Aria Material

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	"string"	–

User_Field

Syntax

Scattering Cross Section [*{of}* *SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Scattering Cross Section [*{of}* *SpeciesName*] = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Scope

Aria Material

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Scattering Cross Section [*{of} SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	“string”	–
plugin_parameters	“string”	–

Schmidt Number

Constant

Syntax

Schmidt Number [*{of} SpeciesName*] = Constant *{sc | value}* = *Sc*

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Sc	real	–

Copied

Syntax

Schmidt Number [*{of}* *SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	“string”	–

Exponential

Syntax

Schmidt Number [*{of}* *SpeciesName*] = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Scope

Aria Material

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Schmidt Number [*{of} SpeciesName*] = Global (Global_Name = *global_name* | [Component = *component*])

Scope

Aria Material

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Schmidt Number [*{of} SpeciesName*] = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Schmidt Number [*{of}* *SpeciesName*] = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Schmidt Number [*{of}* *SpeciesName*] = Scalar_String_Function F = *f*

Scope

Aria Material

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	"string"	–

User_Field

Syntax

Schmidt Number [*{of} SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Schmidt Number [*{of} SpeciesName*] = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Scope

Aria Material

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Schmidt Number [*{of} SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	“string”	–
plugin_parameters	“string”	–

Seebeck Coefficient

Constant

Syntax

Seebeck Coefficient [*{of} SpeciesName*] = Constant *{a | value}* = *a*

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
a	real	–

Copied

Syntax

Seebeck Coefficient [*{of}* *SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	“string”	–

Exponential

Syntax

Seebeck Coefficient [*{of}* *SpeciesName*] = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Scope

Aria Material

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Seebeck Coefficient [*{of} SpeciesName*] = Global (Global_Name = *global_name* | [Component = *component*])

Scope

Aria Material

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Seebeck Coefficient [*{of} SpeciesName*] = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Seebeck Coefficient [*{of}* *SpeciesName*] = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Seebeck Coefficient [*{of}* *SpeciesName*] = Scalar_String_Function F = *f*

Scope

Aria Material

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	"string"	–

User_Field

Syntax

Seebeck Coefficient [*{of}* *SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Seebeck Coefficient [*{of}* *SpeciesName*] = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Scope

Aria Material

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Seebeck Coefficient [*{of} SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	“string”	–
plugin_parameters	“string”	–

Shell Lofting Factor

Constant

Syntax

Shell Lofting Factor [*{of} SpeciesName*] = Constant *{f | value}* = *f*

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	real	–

Copied

Syntax

Shell Lofting Factor [*{of}* *SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	“string”	–

Element_Attribute

Syntax

Shell Lofting Factor [*{of}* *SpeciesName*] = Element_Attribute (Attribute_Name = *attribute_name* | [Multiplier = *multiplier*])

Scope

Aria Material

Summary

Element Distribution Factor for the lofting of shell surface

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
attribute_name	“string”	–
multiplier	real	1

Exponential

Syntax

Shell Lofting Factor [*{of}* *SpeciesName*] = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Scope

Aria Material

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Shell Lofting Factor [*{of}* *SpeciesName*] = Global (Global_Name = *global_name* | [Component = *component*])

Scope

Aria Material

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Shell Lofting Factor [*{of}* *SpeciesName*] = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Shell Lofting Factor [*{of}* *SpeciesName*] = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Shell Lofting Factor [*{of} SpeciesName*] = Scalar_String_Function F = *f*

Scope

Aria Material

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	"string"	–

User_Field

Syntax

Shell Lofting Factor [*{of}* *SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Shell Lofting Factor [*{of}* *SpeciesName*] = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Scope

Aria Material

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Shell Lofting Factor [*{of} SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	“string”	–
plugin_parameters	“string”	–

Shell Thickness

Constant

Syntax

Shell Thickness [*{of} SpeciesName*] = Constant *{t | value}* = *t*

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
t	real	–

Copied

Syntax

Shell Thickness [*{of}* *SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	“string”	–

Element_Attribute

Syntax

Shell Thickness [*{of}* *SpeciesName*] = Element_Attribute ([Contact = *contact*] | [Attribute_Name = *attribute_name*] | [Multiplier = *multiplier*])

Scope

Aria Material

Summary

Distribution Factor for the shell thickness

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
contact	“string”	–
attribute_name	“string”	THICKNESS
multiplier	real	1

Exponential

Syntax

Shell Thickness [*{of}* *SpeciesName*] = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Scope

Aria Material

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Shell Thickness [*{of}* *SpeciesName*] = Global (Global_Name = *global_name* | [Component = *component*])

Scope

Aria Material

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Shell Thickness [*{of}* *SpeciesName*] = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Shell Thickness [*{of}* *SpeciesName*] = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Shell Thickness [*{of} SpeciesName*] = Scalar_String_Function F = *f*

Scope

Aria Material

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	"string"	–

User_Field

Syntax

Shell Thickness [*{of}* *SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Shell Thickness [*{of}* *SpeciesName*] = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Scope

Aria Material

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Shell Thickness [*{of} SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	“string”	–
plugin_parameters	“string”	–

Skeleton Density

Constant

Syntax

Skeleton Density [*{of} SpeciesName*] = Constant *{rho | value}* = *rho*

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
rho	real	–

Copied

Syntax

Skeleton Density [*{of}* *SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	“string”	–

Exponential

Syntax

Skeleton Density [*{of}* *SpeciesName*] = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Scope

Aria Material

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Skeleton Density [*{of}* *SpeciesName*] = Global (Global_Name = *global_name* | [Component = *component*])

Scope

Aria Material

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Skeleton Density [*{of}* *SpeciesName*] = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Skeleton Density [*{of}* *SpeciesName*] = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Skeleton Density [*{of}* *SpeciesName*] = Scalar_String_Function F = *f*

Scope

Aria Material

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	"string"	–

User_Field

Syntax

Skeleton Density [*{of}* *SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Skeleton Density [*{of}* *SpeciesName*] = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Scope

Aria Material

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Skeleton Density [*{of}* *SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	“string”	–
plugin_parameters	“string”	–

Skeleton Enthalpy

Cpt

Syntax

Skeleton Enthalpy [*{of}* *SpeciesName*] = Cpt ()

Scope

Aria Material

Summary

CPT model for the solid skeleton enthalpy

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–

Skeleton Internal Energy

Constant

Syntax

Skeleton Internal Energy [*{of} SpeciesName*] = Constant *{e | value}* = *e*

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
e	real	–

Copied

Syntax

Skeleton Internal Energy [*{of} SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	“string”	–

Cpt

Syntax

Skeleton Internal Energy [*{of} SpeciesName*] = Cpt ()

Scope

Aria Material

Summary

CPT model for the solid skeleton internal energy

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–

Exponential

Syntax

Skeleton Internal Energy [*{of}* *SpeciesName*] = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Scope

Aria Material

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Skeleton Internal Energy [*{of}* *SpeciesName*] = Global (Global_Name = *global_name* | [Component = *component*])

Scope

Aria Material

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
global_name	“string”	–
component	integer	0

Linear

Syntax

Skeleton Internal Energy [*{of}* *SpeciesName*] = Linear (Cp = *cp* | T_Ref = *T_ref*)

Scope

Aria Material

Summary

CPT model for the solid skeleton internal energy

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
cp	real	–
T_ref	real	–

Nd_Table

Syntax

Skeleton Internal Energy [*{of}* *SpeciesName*] = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Skeleton Internal Energy [*{of}* *SpeciesName*] = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Skeleton Internal Energy [*{of} SpeciesName*] = Scalar_String_Function F = *f*

Scope

Aria Material

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	"string"	–

User_Field

Syntax

Skeleton Internal Energy [*{of}* *SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Skeleton Internal Energy [*{of}* *SpeciesName*] = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Scope

Aria Material

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Skeleton Internal Energy [*{of} SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	“string”	–
plugin_parameters	“string”	–

Skeleton Specific Heat

Constant

Syntax

Skeleton Specific Heat [*{of} SpeciesName*] = Constant *{cp | value}* = *cp*

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
cp	real	–

Copied

Syntax

Skeleton Specific Heat [*{of}* *SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	“string”	–

Exponential

Syntax

Skeleton Specific Heat [*{of}* *SpeciesName*] = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Scope

Aria Material

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Skeleton Specific Heat [*{of} SpeciesName*] = Global (Global_Name = *global_name* | [Component = *component*])

Scope

Aria Material

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Skeleton Specific Heat [*{of} SpeciesName*] = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Skeleton Specific Heat [*{of}* *SpeciesName*] = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Skeleton Specific Heat [*{of}* *SpeciesName*] = Scalar_String_Function F = *f*

Scope

Aria Material

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	"string"	–

User_Field

Syntax

Skeleton Specific Heat [*{of} SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Skeleton Specific Heat [*{of} SpeciesName*] = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Scope

Aria Material

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Skeleton Specific Heat [*{of} SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	“string”	–
plugin_parameters	“string”	–

Solid Body Acceleration

Constant

Syntax

Solid Body Acceleration [*{of} SpeciesName*] = Constant ([Valuex = *valuex*] | [Valuey = *valuey*] | [Valuez = *valuez*])

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
valuex	real	0
valuey	real	0
valuez	real	0

Copied

Syntax

Solid Body Acceleration [*{of}* *SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	“string”	–

Scalar_Vector_Product

Syntax

Solid Body Acceleration [*{of}* *SpeciesName*] = Scalar_Vector_Product (Vector = *vector* | [Scalars = *scalars*] | [Multiplier = *multiplier*])

Scope

Aria Material

Summary

Scalar vector product

Description

Multiplies a vector by a list of provided scalars (variables) and a multiplier (constant)

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
vector	“string”	–
scalars	“string”	–
multiplier	real	1

Tensor_Vector_Product

Syntax

Solid Body Acceleration [*{of} SpeciesName*] = Tensor_Vector_Product (Vector = *vector* | Tensor = *tensor* | [Scale = *scale*])

Scope

Aria Material

Summary

Tensor vector product

Description

Calculates a vector resulting from a tensor-vector product with an optional scaling factor.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
vector	“string”	–
tensor	“string”	–
scale	real	1

User_Field

Syntax

Solid Body Acceleration [*{of} SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Vector from User Field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
scaling	real	1
global_var	“string”	–
time_function	“string”	–

User_Function

Syntax

Solid Body Acceleration [*{of} SpeciesName*] = User_Function ([Name_X = *name_x*] | [Name_Y = *name_y*] | [Name_Z = *name_z*] | X = *x* | [Multiplier = *multiplier*])

Scope

Aria Material

Summary

Value from a Vector User Function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name_x	“string”	–
name_y	“string”	–
name_z	“string”	–
x	“string”	–
multiplier	real	1

User_Plugin

Syntax

Solid Body Acceleration [*{of} SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	“string”	–
plugin_parameters	“string”	–

Vector_String_Function

Syntax

Solid Body Acceleration [*{of}* *SpeciesName*] = Vector_String_Function (F_X = *f_x* | F_Y = *f_y* | [F_Z = *f_z*])

Scope

Aria Material

Summary

Value from a string function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f_x	“string”	–
f_y	“string”	–
f_z	“string”	–

Solid Density

Constant

Syntax

Solid Density [*{of}* *SpeciesName*] = Constant *{rho_solid | value}* = *rho_solid*

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
rho_solid	real	–

Copied

Syntax

Solid Density [*{of}* *SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	"string"	–

Exponential

Syntax

Solid Density [*{of}* *SpeciesName*] = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Scope

Aria Material

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Solid Density [*{of} SpeciesName*] = Global (Global_Name = *global_name* | [Component = *component*])

Scope

Aria Material

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
global_name	“string”	–
component	integer	0

Nd_Table

Syntax

Solid Density [*{of} SpeciesName*] = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Solid Density [*{of}* *SpeciesName*] = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Solid Density [*{of}* *SpeciesName*] = Scalar_String_Function F = *f*

Scope

Aria Material

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be

a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	"string"	–

User_Field

Syntax

Solid Density [*{of}* *SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Solid Density [*{of}* *SpeciesName*] = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Scope

Aria Material

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1
toggle	"string"	–

User_Plugin

Syntax

Solid Density [*{of} SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	"string"	–
plugin_parameters	"string"	–

Solid Displacements

Constant

Syntax

Solid Displacements [*{of} SpeciesName*] = Constant ([X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
x	real	0
y	real	0
z	real	0

Copied

Syntax

Solid Displacements [*{of}* *SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	“string”	–

User_Field

Syntax

Solid Displacements [*{of}* *SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Vector from User Field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
scaling	real	1
global_var	“string”	–
time_function	“string”	–

User_Function

Syntax

Solid Displacements [*{of} SpeciesName*] = User_Function ([Name_X = *name_x*] | [Name_Y = *name_y*] | [Name_Z = *name_z*] | X = *x* | [Multiplier = *multiplier*])

Scope

Aria Material

Summary

Value from a Vector User Function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name_x	"string"	–
name_y	"string"	–
name_z	"string"	–
x	"string"	–
multiplier	real	1

User_Plugin

Syntax

Solid Displacements [*{of} SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	"string"	–
plugin_parameters	"string"	–

Vector_String_Function

Syntax

Solid Displacements [*{of}* *SpeciesName*] = Vector_String_Function (F_X = *f_x* | F_Y = *f_y* | [F_Z = *f_z*])

Scope

Aria Material

Summary

Value from a string function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f_x	"string"	–
f_y	"string"	–
f_z	"string"	–

Solid Displacements X

Constant

Syntax

Solid Displacements X [*{of}* *SpeciesName*] = Constant Value = *value*

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
value	real	–

Copied

Syntax

Solid Displacements X [*{of}* *SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	"string"	–

Exponential

Syntax

Solid Displacements X [*{of}* *SpeciesName*] = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Scope

Aria Material

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Solid Displacements X [*{of} SpeciesName*] = Global (Global_Name = *global_name* | [Component = *component*])

Scope

Aria Material

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
global_name	“string”	–
component	integer	0

Nd_Table

Syntax

Solid Displacements X [*{of} SpeciesName*] = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Solid Displacements X [*{of}* *SpeciesName*] = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Solid Displacements X [*{of}* *SpeciesName*] = Scalar_String_Function F = *f*

Scope

Aria Material

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be

a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	"string"	–

User_Field

Syntax

Solid Displacements X [*{of}* *SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Solid Displacements X [*{of}* *SpeciesName*] = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Scope

Aria Material

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The

meaning of each option is as follows:

Toggle
If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1
toggle	"string"	–

User_Plugin

Syntax
Solid Displacements X [*{of} SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope
Aria Material

Summary
Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	"string"	–
plugin_parameters	"string"	–

Solid Displacements Y

Constant

Syntax
Solid Displacements Y [*{of} SpeciesName*] = Constant Value = *value*

Scope
Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
value	real	–

Copied

Syntax

Solid Displacements Y [*{of}* *SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	“string”	–

Exponential

Syntax

Solid Displacements Y [*{of}* *SpeciesName*] = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Scope

Aria Material

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Solid Displacements Y [*{of}* *SpeciesName*] = Global (Global_Name = *global_name* | [Component = *component*])

Scope

Aria Material

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Solid Displacements Y [*{of}* *SpeciesName*] = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Solid Displacements Y [*{of} SpeciesName*] = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Solid Displacements Y [*{of}* *SpeciesName*] = Scalar_String_Function F = *f*

Scope

Aria Material

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	"string"	–

User_Field

Syntax

Solid Displacements Y [*{of} SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Solid Displacements Y [*{of} SpeciesName*] = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Scope

Aria Material

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Solid Displacements Y [*{of} SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	“string”	–
plugin_parameters	“string”	–

Solid Displacements Z

Constant

Syntax

Solid Displacements Z [*{of} SpeciesName*] = Constant Value = *value*

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
value	real	–

Copied

Syntax

Solid Displacements Z [*{of}* *SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	“string”	–

Exponential

Syntax

Solid Displacements Z [*{of}* *SpeciesName*] = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Scope

Aria Material

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Solid Displacements Z [*{of}* *SpeciesName*] = Global (Global_Name = *global_name* | [Component = *component*])

Scope

Aria Material

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Solid Displacements Z [*{of}* *SpeciesName*] = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Solid Displacements Z [*{of}* *SpeciesName*] = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Solid Displacements Z [*{of} SpeciesName*] = Scalar_String_Function F = *f*

Scope

Aria Material

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	"string"	–

User_Field

Syntax

Solid Displacements Z [*{of} SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Solid Displacements Z [*{of} SpeciesName*] = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Scope

Aria Material

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Solid Displacements Z [*{of} SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	“string”	–
plugin_parameters	“string”	–

Solid Lambda

Constant

Syntax

Solid Lambda [*{of} SpeciesName*] = Constant *{lambda | value}* = *lambda*

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
lambda	real	–

Converted

Syntax

Solid Lambda [*{of}* *SpeciesName*] = Converted ()

Scope

Aria Material

Summary

Lame coefficient computed from Poisson's ratio and elastic modulus

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–

Converted_Plane_Stress

Syntax

Solid Lambda [*{of}* *SpeciesName*] = Converted_Plane_Stress ()

Scope

Aria Material

Summary

This model has been deprecated and removed from the code.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–

Copied

Syntax

Solid Lambda [*{of}* *SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	"string"	–

Exponential

Syntax

Solid Lambda [*{of}* *SpeciesName*] = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Scope

Aria Material

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Solid Lambda [*{of} SpeciesName*] = Global (Global_Name = *global_name* | [Component = *component*])

Scope

Aria Material

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
global_name	“string”	–
component	integer	0

Nd_Table

Syntax

Solid Lambda [*{of} SpeciesName*] = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Solid Lambda [*{of}* *SpeciesName*] = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Solid Lambda [*{of}* *SpeciesName*] = Scalar_String_Function F = *f*

Scope

Aria Material

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be

a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	"string"	–

User_Field

Syntax

Solid Lambda [*{of} SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Solid Lambda [*{of} SpeciesName*] = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Scope

Aria Material

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1
toggle	"string"	–

User_Plugin

Syntax

Solid Lambda [*{of} SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	"string"	–
plugin_parameters	"string"	–

Solid Poissons Ratio

Constant

Syntax

Solid Poissons Ratio [*{of} SpeciesName*] = Constant *{nu | value}* = *nu*

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
nu	real	–

Copied

Syntax

Solid Poissons Ratio [*{of}* *SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	“string”	–

Exponential

Syntax

Solid Poissons Ratio [*{of}* *SpeciesName*] = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Scope

Aria Material

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Solid Poissons Ratio [*{of} SpeciesName*] = Global (Global_Name = *global_name* | [Component = *component*])

Scope

Aria Material

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Solid Poissons Ratio [*{of} SpeciesName*] = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Solid Poissons Ratio [*{of}* *SpeciesName*] = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Solid Poissons Ratio [*{of}* *SpeciesName*] = Scalar_String_Function F = *f*

Scope

Aria Material

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	"string"	–

User_Field

Syntax

Solid Poissons Ratio [*{of}* *SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Solid Poissons Ratio [*{of}* *SpeciesName*] = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Scope

Aria Material

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Solid Poissons Ratio [*{of} SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	“string”	–
plugin_parameters	“string”	–

Solid Pressure Gradient

Gidaspow

Syntax

Solid Pressure Gradient [*{of} SpeciesName*] = Gidaspow ([G0 = *g0*] | [C = *c*] | [Fmax = *fmax*])

Scope

Aria Material

Summary

This model has been deprecated and removed from the code.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
g0	real	1
c	real	100
fmax	real	0.62

Solid Stress

Constant

Syntax

Solid Stress [*{of}* *SpeciesName*] = Constant ([Xx = *xx*] | [Xy = *xy*] | [Xz = *xz*] | [Yx = *yx*] | [Yy = *yy*] | [Yz = *yz*] | [Zx = *zx*] | [Zy = *zy*] | [Zz = *zz*] | [T11 = *t11*] | [T12 = *t12*] | [T13 = *t13*] | [T21 = *t21*] | [T22 = *t22*] | [T23 = *t23*] | [T31 = *t31*] | [T32 = *t32*] | [T33 = *t33*])

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
xx	real	0
xy	real	0
xz	real	0
yx	real	0
yy	real	0
yz	real	0
zx	real	0
zy	real	0
zz	real	0
t11	real	0
t12	real	0
t13	real	0
t21	real	0
t22	real	0
t23	real	0
t31	real	0
t32	real	0
t33	real	0

Incompressible_Newtonian

Syntax

Solid Stress [*{of}* *SpeciesName*] = Incompressible_Newtonian ([Pressure = *pressure*] | [Phase = *phase*] | [Biot = *biot*] | [Reference_Frame = *reference_frame*])

Scope

Aria Material

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
pressure	“string”	–
phase	“string”	–
biot	real	1
reference_frame	“string”	MOVING

Isothermal

Syntax

Solid Stress [*{of} SpeciesName*] = Isothermal (T = *t* | T_Ref = *t_ref*)

Scope

Aria Material

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
t	real	–
t_ref	real	–

Lame

Syntax

Solid Stress [*{of} SpeciesName*] = Lame Model = *model*

Scope

Aria Material

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
model	“string”	–

Linear_Elastic

Syntax

Solid Stress [*{of} SpeciesName*] = Linear_Elastic [Reference_Frame = *reference_frame*]

Scope

Aria Material

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
reference_frame	“string”	MOVING

Mooney_Rivlin

Syntax

Solid Stress [*{of}* *SpeciesName*] = Mooney_Rivlin ()

Scope

Aria Material

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–

Neohookean_Elastic

Syntax

Solid Stress [*{of}* *SpeciesName*] = Neohookean_Elastic ()

Scope

Aria Material

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–

Nonlinear_Elastic

Syntax

Solid Stress [*{of}* *SpeciesName*] = Nonlinear_Elastic ()

Scope

Aria Material

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–

Porous_Effective

Syntax

Solid Stress [*{of}* *SpeciesName*] = Porous_Effective ([Pressure = *pressure*] | [Phase = *phase*] | [Biot = *biot*])

Scope

Aria Material

Summary

Apply the pressure as an isotropic mesh stress. Primarily used as an effective stress in porous flow problems.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
pressure	"string"	–
phase	"string"	–
biot	real	1

Residual

Syntax

Solid Stress [*{of}* *SpeciesName*] = Residual ([Sx = *sx*] | [Sy = *sy*] | [Sz = *sz*] | [Sxx = *sxx*] | [Sxy = *sxy*] | [Sxz = *sxz*] | [Syy = *syy*] | [Syz = *syz*] | [Szz = *szz*])

Scope

Aria Material

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
sx	real	0
sy	real	0
sz	real	0
sxx	real	0
sxy	real	0
sxz	real	0
syy	real	0
syz	real	0
szz	real	0

Saturation_Weighted_Porous_Effective

Syntax

Solid Stress [*{of}* *SpeciesName*] = Saturation_Weighted_Porous_Effective ([Pressure = *pressure*] | [Phase = *phase*] | [Biot = *biot*])

Scope

Aria Material

Summary

Apply the pressure as an isotropic mesh stress. Primarily used as an effective stress in porous flow problems.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
pressure	"string"	–
phase	"string"	–
biot	real	1

Species

Syntax

Solid Stress [*{of}* *SpeciesName*] = Species (Sref = *sref* | Phase = *phase* | Species = *species*)

Scope

Aria Material

Summary

A linear mechanical strain due to a species concentration

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
sref	real	–
phase	"string"	–
species	"string"	–

Species_Anisotropic

Syntax

Solid Stress [*{of}* *SpeciesName*] = Species_Anisotropic (Sref = *sref* | [Phase = *phase*] | [Species = *species*])

Scope

Aria Material

Summary

Anisotropic linear mechanical strain due to a species concentration

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
sref	real	–
phase	“string”	–
species	“string”	–

Species_Density

Syntax

Solid Stress [*{of}* *SpeciesName*] = Species_Density ()

Scope

Aria Material

Summary

A linear isotropic mechanical strain due to the expansion/contraction of the material due to changes in species concentration or species density changes.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–

Species_Level_Set_Growth

Syntax

Solid Stress [*{of}* *SpeciesName*] = Species_Level_Set_Growth ()

Scope

Aria Material

Summary

A linear mechanical strain due to the expansion/contraction of the material due to changes in species concentration or species density changes. Growth only occurs in the direction of the level set normal.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–

Species_Transversely_Isotropic

Syntax

Solid Stress [*{of}* *SpeciesName*] = Species_Transversely_Isotropic (Refval = *refval* | Beta_N = *beta_n* | Beta_T = *beta_t* | [N_X = *n_x*] | [N_Y = *n_y*] | [N_Z = *n_z*] | [Phase = *phase*] | [Species = *species*])

Scope

Aria Material

Summary

A linear mechanical strain that is transversely isotropic due to a species concentration

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
refval	real	–
beta_n	real	–
beta_t	real	–
n_x	real	0
n_y	real	0
n_z	real	0
phase	“string”	–
species	“string”	–

Tensor_String_Function

Syntax

Solid Stress [*{of} SpeciesName*] = Tensor_String_Function (F_Xx = *f_xx* | F_Yy = *f_yy* | [F_Xy = *f_xy*] | [F_Zz = *f_zz*] | [F_Xz = *f_xz*] | [F_Yz = *f_yz*] | [F_Yx = *f_yx*] | [F_Zx = *f_zx*] | [F_Zy = *f_zy*])

Scope

Aria Material

Summary

Value from a string function.

Description

The tensor components are calculated from the user-defined string functions, which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used. Off-diagonal terms are optional and default to 0 if omitted.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f_xx	"string"	–
f_yy	"string"	–
f_xy	"string"	0.0
f_zz	"string"	–
f_xz	"string"	0.0
f_yz	"string"	0.0
f_yx	"string"	0.0
f_zx	"string"	0.0
f_zy	"string"	0.0

Tensor_String_Function_Symmetric

Syntax

Solid Stress [*{of} SpeciesName*] = Tensor_String_Function_Symmetric (F_Xx = *f_xx* | F_Yy = *f_yy* | [F_Xy = *f_xy*] | [F_Zz = *f_zz*] | [F_Xz = *f_xz*] | [F_Yz = *f_yz*])

Scope

Aria Material

Summary

Value from a string function.

Description

The symmetric tensor components are calculated from the user-defined string functions (only for diagonal and upper triangular terms), which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used. Off-diagonal terms are optional and default to 0 if omitted.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f_xx	"string"	–
f_yy	"string"	–
f_xy	"string"	0.0
f_zz	"string"	–
f_xz	"string"	0.0
f_yz	"string"	0.0

Thermal

Syntax

Solid Stress [*{of}* *SpeciesName*] = Thermal ([T_Ref = *T_ref*] | [Beta = *beta*])

Scope

Aria Material

Summary

A linear mechanical strain due to a temperature change from reference temperature.

$$\sigma_{xx} = \sigma_{yy} = \sigma_{zz} = beta(T - T_{ref})$$

where β is a thermal expansion coefficient and T_{ref} is some reference temperature. Alternatively, beta can also be specified as a model in the material

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
T_ref	real	0
beta	real	0

User_Plugin

Syntax

Solid Stress [*{of} SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	“string”	–
plugin_parameters	“string”	–

Solid Two Mu

Constant

Syntax

Solid Two Mu [*{of} SpeciesName*] = Constant *{two_mu | value}* = *two_mu*

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
two_mu	real	–

Converted

Syntax

Solid Two Mu [*{of}* *SpeciesName*] = Converted ()

Scope

Aria Material

Summary

Lame coefficient computed from Poisson's ratio and elastic modulus

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–

Copied

Syntax

Solid Two Mu [*{of}* *SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	"string"	–

Exponential

Syntax

Solid Two Mu [*{of}* *SpeciesName*] = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Scope

Aria Material

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the

constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Solid Two Mu [*{of} SpeciesName*] = Global (Global_Name = *global_name* | [Component = *component*])

Scope

Aria Material

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Solid Two Mu [*{of} SpeciesName*] = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Solid Two Mu [*{of}* *SpeciesName*] = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Solid Two Mu [*{of} SpeciesName*] = Scalar_String_Function F = *f*

Scope

Aria Material

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	"string"	–

User_Field

Syntax

Solid Two Mu [*{of} SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Solid Two Mu [*{of} SpeciesName*] = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Scope

Aria Material

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Solid Two Mu [*{of}* *SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	“string”	–
plugin_parameters	“string”	–

Solid Velocity

Constant

Syntax

Solid Velocity [*{of}* *SpeciesName*] = Constant ([X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
x	real	0
y	real	0
z	real	0

Copied

Syntax

Solid Velocity [*{of}* *SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	"string"	–

User_Field

Syntax

Solid Velocity [*{of}* *SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Vector from User Field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Solid Velocity [*{of}* *SpeciesName*] = User_Function ([Name_X = *name_x*] | [Name_Y = *name_y*] | [Name_Z = *name_z*] | X = *x* | [Multiplier = *multiplier*])

Scope

Aria Material

Summary

Value from a Vector User Function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name_x	"string"	–
name_y	"string"	–
name_z	"string"	–
x	"string"	–
multiplier	real	1

User_Plugin

Syntax

Solid Velocity [*{of}* *SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	"string"	–
plugin_parameters	"string"	–

Vector_String_Function

Syntax

Solid Velocity [*{of}* *SpeciesName*] = Vector_String_Function (F_X = *f_x* | F_Y = *f_y* | [F_Z = *f_z*])

Scope

Aria Material

Summary

Value from a string function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f_x	"string"	–
f_y	"string"	–
f_z	"string"	–

Solid Velocity X

Constant

Syntax

Solid Velocity X [*{of}* *SpeciesName*] = Constant Value = *value*

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
value	real	–

Copied

Syntax

Solid Velocity X [*{of}* *SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	"string"	–

Exponential

Syntax

Solid Velocity X [*{of}* *SpeciesName*] = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Scope

Aria Material

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Solid Velocity X [*{of}* *SpeciesName*] = Global (Global_Name = *global_name* | [Component = *component*])

Scope

Aria Material

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Solid Velocity X [*{of}* *SpeciesName*] = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
vars	"string"1["string"2["string"3["string"4["string"5]]]]	–
i0	"string"	–
i1	"string"	–
i2	"string"	–
x	"string"	–
y	"string"	–
z	"string"	–

Polynomial

Syntax

Solid Velocity X [*{of}* *SpeciesName*] = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Solid Velocity X [*{of}* *SpeciesName*] = Scalar_String_Function F = *f*

Scope

Aria Material

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be

a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	"string"	–

User_Field

Syntax

Solid Velocity X [*{of}* *SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Solid Velocity X [*{of}* *SpeciesName*] = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Scope

Aria Material

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1
toggle	"string"	–

User_Plugin

Syntax

Solid Velocity X [*{of} SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	"string"	–
plugin_parameters	"string"	–

Solid Velocity Y

Constant

Syntax

Solid Velocity Y [*{of} SpeciesName*] = Constant Value = *value*

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
value	real	–

Copied

Syntax

Solid Velocity Y [*{of}* *SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	"string"	–

Exponential

Syntax

Solid Velocity Y [*{of}* *SpeciesName*] = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Scope

Aria Material

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Solid Velocity Y [*{of} SpeciesName*] = Global (Global_Name = *global_name* | [Component = *component*])

Scope

Aria Material

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Solid Velocity Y [*{of} SpeciesName*] = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Solid Velocity Y [*{of}* *SpeciesName*] = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Solid Velocity Y [*{of} SpeciesName*] = Scalar_String_Function F = *f*

Scope

Aria Material

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	"string"	–

User_Field

Syntax

Solid Velocity Y [*{of}* *SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Solid Velocity Y [*{of}* *SpeciesName*] = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Scope

Aria Material

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Solid Velocity Y [*{of} SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	“string”	–
plugin_parameters	“string”	–

Solid Velocity Z

Constant

Syntax

Solid Velocity Z [*{of} SpeciesName*] = Constant Value = *value*

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
value	real	–

Copied

Syntax

Solid Velocity Z [*{of}* *SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	"string"	–

Exponential

Syntax

Solid Velocity Z [*{of}* *SpeciesName*] = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Scope

Aria Material

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Solid Velocity Z [*{of} SpeciesName*] = Global (Global_Name = *global_name* | [Component = *component*])

Scope

Aria Material

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Solid Velocity Z [*{of} SpeciesName*] = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Solid Velocity Z [*{of}* *SpeciesName*] = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Solid Velocity Z [*{of} SpeciesName*] = Scalar_String_Function F = *f*

Scope

Aria Material

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	"string"	–

User_Field

Syntax

Solid Velocity Z [*{of} SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Solid Velocity Z [*{of} SpeciesName*] = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Scope

Aria Material

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1
toggle	"string"	–

User_Plugin

Syntax

Solid Velocity Z [*{of} SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	"string"	–
plugin_parameters	"string"	–

Solid Youngs Modulus

Constant

Syntax

Solid Youngs Modulus [*{of} SpeciesName*] = Constant *{e | value}* = *e*

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
e	real	–

Copied

Syntax

Solid Youngs Modulus [*{of}* *SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	"string"	–

Exponential

Syntax

Solid Youngs Modulus [*{of}* *SpeciesName*] = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Scope

Aria Material

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Solid Youngs Modulus [*{of} SpeciesName*] = Global (Global_Name = *global_name* | [Component = *component*])

Scope

Aria Material

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Solid Youngs Modulus [*{of} SpeciesName*] = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Solid Youngs Modulus [*{of}* *SpeciesName*] = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Solid Youngs Modulus [*{of} SpeciesName*] = Scalar_String_Function F = *f*

Scope

Aria Material

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	"string"	–

User_Field

Syntax

Solid Youngs Modulus [*{of}* *SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Solid Youngs Modulus [*{of}* *SpeciesName*] = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Scope

Aria Material

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Solid Youngs Modulus [*{of} SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	“string”	–
plugin_parameters	“string”	–

Solidus Temperature

Constant

Syntax

Solidus Temperature [*{of} SpeciesName*] = Constant Value = *value*

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
value	real	–

Copied

Syntax

Solidus Temperature [*{of}* *SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	“string”	–

Exponential

Syntax

Solidus Temperature [*{of}* *SpeciesName*] = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Scope

Aria Material

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Solidus Temperature [*{of}* *SpeciesName*] = Global (Global_Name = *global_name* | [Component = *component*])

Scope

Aria Material

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Solidus Temperature [*{of}* *SpeciesName*] = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Solidus Temperature [*{of}* *SpeciesName*] = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Solidus Temperature [*{of}* *SpeciesName*] = Scalar_String_Function F = *f*

Scope

Aria Material

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	"string"	–

User_Field

Syntax

Solidus Temperature [*{of} SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Solidus Temperature [*{of} SpeciesName*] = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Scope

Aria Material

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Solidus Temperature [*{of} SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	“string”	–
plugin_parameters	“string”	–

Soret Coefficient

Constant

Syntax

Soret Coefficient [*{of} SpeciesName*] = Constant *{s | value}* = *s*

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
s	real	–

Copied

Syntax

Soret Coefficient [*{of}* *SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	“string”	–

Exponential

Syntax

Soret Coefficient [*{of}* *SpeciesName*] = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Scope

Aria Material

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Soret Coefficient [*{of} SpeciesName*] = Global (Global_Name = *global_name* | [Component = *component*])

Scope

Aria Material

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Soret Coefficient [*{of} SpeciesName*] = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Soret Coefficient [*{of} SpeciesName*] = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Soret Coefficient [*{of} SpeciesName*] = Scalar_String_Function F = *f*

Scope

Aria Material

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	"string"	–

User_Field

Syntax

Soret Coefficient [*{of} SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Soret Coefficient [*{of} SpeciesName*] = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Scope

Aria Material

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Soret Coefficient [*{of} SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	“string”	–
plugin_parameters	“string”	–

Sound Speed

Constant

Syntax

Sound Speed [*{of} SpeciesName*] = Constant *{c | value}* = *c*

Scope

Aria Material

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
c	real	–

Species

Chemeq_Gas

Syntax

Species [*{of}* *SpeciesName*] = Chemeq_Gas ()

Scope

Aria Material

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–

Constant

Syntax

Species [*{of}* *SpeciesName*] = Constant Value = *value*

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
value	real	–

Copied

Syntax

Species [*{of}* *SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	"string"	–

Exponential

Syntax

Species [*{of}* *SpeciesName*] = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Scope

Aria Material

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

From_Chemeq

Syntax

Species [*{of}* *SpeciesName*] = From_Chemeq ()

Scope

Aria Material

Summary

Species concentration of one species extracted from ChemEq

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–

From_Density

Syntax

Species [*{of}* *SpeciesName*] = From_Density ()

Scope

Aria Material

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–

From_Mass_Fraction

Syntax

Species [*{of}* *SpeciesName*] = From_Mass_Fraction ()

Scope

Aria Material

Summary

Molar concentration of a single species calculated from its mass fraction, molecular weight, and overall density

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–

From_Material_Phase

Syntax

Species [*{of}* *SpeciesName*] = From_Material_Phase Phase_Name = *phase_name*

Scope

Aria Material

Summary

Copy the species from another material phase to the desired material phase.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
phase_name	"string"	–

From_Phase_All

Syntax

Species [*{of}* *SpeciesName*] = From_Phase_All ()

Scope

Aria Material

Summary

Copy the species from PHASE_ALL to the desired phase.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–

From_Species_Fraction

Syntax

Species [*{of}* *SpeciesName*] = From_Species_Fraction ()

Scope

Aria Material

Summary

Molar concentration of a single species calculated from its mole (species) fraction and overall molar density

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–

Global

Syntax

Species [*{of}* *SpeciesName*] = Global (Global_Name = *global_name* | [Component = *component*])

Scope

Aria Material

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
global_name	“string”	–
component	integer	0

Nd_Table

Syntax

Species [*{of}* *SpeciesName*] = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Partial_Molar_Volume

Syntax

Species [*{of}* *SpeciesName*] = Partial_Molar_Volume ()

Scope

Aria Material

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–

Phase_Average

Syntax

Species [*{of}* *SpeciesName*] = Phase_Average ([Subindex_A = *subindex_a*] | [Subindex_B = *subindex_b*])

Scope

Aria Material

Summary

Phase-averaged values

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
subindex_a	integer	-1
subindex_b	integer	-1

Polynomial

Syntax

Species [*{of}* *SpeciesName*] = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Species [*{of}* *SpeciesName*] = Scalar_String_Function F = *f*

Scope

Aria Material

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	"string"	–

Solvent_Concentration_From_Solute_Concentrations_And_Partial_Molar_Volumes

Syntax

Species [*{of}* *SpeciesName*] = Solvent_Concentration_From_Solute_Concentrations_And_Partial_Molar_Volumes ()

Scope

Aria Material

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–

Sum_All_Species

Syntax

Species [*{of}* *SpeciesName*] = Sum_All_Species ()

Scope

Aria Material

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–

User_Field

Syntax

Species [*{of}* *SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Species [*{of}* *SpeciesName*] = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Scope

Aria Material

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1
toggle	"string"	–

User_Plugin

Syntax

Species [*{of}* *SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	"string"	–
plugin_parameters	"string"	–

Species Diffusion

All_Subindices_Use_Stefan_Maxwell

Syntax

Species Diffusion [*{of}* *SpeciesName*] = All_Subindices_Use_Stefan_Maxwell (Velocityconversion = *velocityConversion* | Canteraxmlfile = *canteraXMLFile* | [Solidscaling = *solidScaling*] | [Velocitybasis = *velocityBasis*])

Scope

Aria Material

Summary

Computes the diffusive flux for all species present in the problem using Cantera's Stefan-Maxwell model.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
velocityConversion	real	–
canteraXMLFile	"string"	–
solidScaling	real	1e-12
velocityBasis	"string"	MOLE_AVERAGE

Basic

Syntax

Species Diffusion [*{of}* *SpeciesName*] = Basic ()

Scope

Aria Material

Summary

Generalized model for species diffusion - can use either a scalar or tensor species diffusivity.

Description

If a tensor species diffusivity D_i is provided in the material model, the form of flux is

$$\mathbf{J} = -D_i \cdot \nabla S_i$$

If the species diffusivity is given as a scalar d_i then the heat flux is

$$\mathbf{J} = -d_i \nabla S_i$$

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–

Chemical_Potential

Syntax

Species Diffusion [*{of}* *SpeciesName*] = Chemical_Potential ()

Scope

Aria Material

Summary

Model species diffusion using a flux determined from the gradient of the chemical potential.$\vec{J} = -Mc\nabla\mu$ where M is the mobility (either scalar or tensor), c the species concentration, and mu the chemical potential.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–

Constant

Syntax

Species Diffusion [*{of}* *SpeciesName*] = Constant ([Valuex = *valuex*] | [Valuey = *valuey*] | [Valuez = *valuez*])

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
valuex	real	0
valuey	real	0
valuez	real	0

Constant_Tensorial

Syntax

Species Diffusion [*{of}* *SpeciesName*] = Constant_Tensorial ([D_Xx = *d_xx*] | [D_Xy = *d_xy*] | [D_Xz = *d_xz*] | [D_Yx = *d_yx*] | [D_Yy = *d_yy*] | [D_Yz = *d_yz*] | [D_Zx = *d_zx*] | [D_Zy = *d_zy*] | [D_Zz = *d_zz*] | [D_T11 = *d_t11*] | [D_T12 = *d_t12*] | [D_T13 = *d_t13*] | [D_T21 = *d_t21*] | [D_T22 = *d_t22*] | [D_T23 = *d_t23*] | [D_T31 = *d_t31*] | [D_T32 = *d_t32*] | [D_T33 = *d_t33*])

Scope

Aria Material

Summary

Tensor species diffusion model where the constant species diffusivity tensor parameters are provided.

Description

The form of flux is

$$\mathbf{J} = -D_i \cdot \nabla S_i$$

where D_i is a constant tensor defined via the provided model parameters.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
d_xx	real	0
d_xy	real	0
d_xz	real	0
d_yx	real	0
d_yy	real	0
d_yz	real	0
d_zx	real	0
d_zy	real	0
d_zz	real	0
d_t11	real	0
d_t12	real	0
d_t13	real	0
d_t21	real	0
d_t22	real	0
d_t23	real	0
d_t31	real	0
d_t32	real	0
d_t33	real	0

Copied

Syntax

Species Diffusion [*{of}* *SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	“string”	–

Darcy

Syntax

Species Diffusion [*{of}* *SpeciesName*] = Darcy ()

Scope

Aria Material

Description

The form of flux is

$$\mathbf{J} = C_i \mathbf{v}$$

where C_i is the species concentration and $\mathbf{v}$ the velocity.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–

Drift_Diffusion

Syntax

Species Diffusion [*{of}* *SpeciesName*] = Drift_Diffusion ()

Scope

Aria Material

Summary

Drift diffusion model for charged plasma species.

Description

The flux takes on the following form, $\mathbf{J} = -D\nabla C - z\mu C\nabla V$, where D is the species diffusivity, C the species concentration, μ the species mobility, z the species valence, and $-\nabla V$ the electric field.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–

Drift_Diffusion_Magnetic

Syntax

Species Diffusion [*{of}* *SpeciesName*] = Drift_Diffusion_Magnetic ()

Scope

Aria Material

Summary

Drift diffusion model for charged plasma species with magnetic field.

Description

The flux takes on the following form, $\mathbf{J} = -D\nabla C - z\mu C\nabla V - \mu^2 C\nabla V \times \mathbf{B}$, where D is the species diffusivity, C the species concentration, μ the species mobility, z the species valence, $-\nabla V$ the electric field, and $\mathbf{B}$ the magnetic field.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–

Ficks_Law

Syntax

Species Diffusion [*{of}* *SpeciesName*] = Ficks_Law ()

Scope

Aria Material

Summary

Generalized model for species diffusion - can use either a scalar or tensor species diffusivity. Identical to BASIC model.

Description

If a tensor species diffusivity D_i is provided in the material model, the form of flux is

$$\mathbf{J} = -D_i \cdot \nabla S_i$$

If the species diffusivity is given as a scalar d_i then the heat flux is

$$\mathbf{J} = -d_i \nabla S_i$$

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–

Mass_Balance_Fracbal

Syntax

Species Diffusion [*{of}* *SpeciesName*] = Mass_Balance_Fracbal ()

Scope

Aria Material

Summary

Species diffusion for the fracbal species in a mass balance equation system. $\vec{J}_i = -1 \sum_{j \neq i} (m\vec{d}ot_j)/MW_i$.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–

Nernst_Planck

Syntax

Species Diffusion [*{of}* *SpeciesName*] = Nernst_Planck ([F = *f*] | [R = *r*])

Scope

Aria Material

Description

The diffusive flux is

$$\mathbf{J} = C\left(\frac{v_i S_i F D_i}{RT}\nabla V - D\nabla S_i\right)$$

where v_i is the valence of the ion species, S_i the molar concentration, F the provided Faraday's constant, D_i the SPECIES_DIFFUSIVITY, R the provided gas constant, T the temperature, and V the voltage.

If both POROSITY and TORTUOSITY_FACTOR models are present then $C = \phi\tau$, else $C = 1$.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	real	9.649e+04
r	real	8.314

Scalar_Vector_Product

Syntax

Species Diffusion [*{of} SpeciesName*] = Scalar_Vector_Product (Vector = *vector* | [Scalars = *scalars*] | [Multiplier = *multiplier*])

Scope

Aria Material

Summary

Scalar vector product

Description

Multiplies a vector by a list of provided scalars (variables) and a multiplier (constant)

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
vector	"string"	–
scalars	"string"	–
multiplier	real	1

Tensor_Chemical_Potential

Syntax

Species Diffusion [*{of} SpeciesName*] = Tensor_Chemical_Potential ()

Scope

Aria Material

Summary

Same as CHEMICAL_POTENTIAL model. Model species diffusion using a flux determined from the gradient of the chemical potential.$\vec{J} = -Mc\nabla\mu$ where M is the mobility (either scalar or tensor), c the species concentration, and mu the chemical potential.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–

Tensor_Thermophoresis

Syntax

Species Diffusion [*{of}* *SpeciesName*] = Tensor_Thermophoresis ()

Scope

Aria Material

Summary

This model has been deprecated and removed from the code.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–

Tensor_Vector_Product

Syntax

Species Diffusion [*{of}* *SpeciesName*] = Tensor_Vector_Product (Vector = *vector* | Tensor = *tensor* | [Scale = *scale*])

Scope

Aria Material

Summary

Tensor vector product

Description

Calculates a vector resulting from a tensor-vector product with an optional scaling factor.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
vector	“string”	–
tensor	“string”	–
scale	real	1

Tensorial

Syntax

Species Diffusion [*{of}* *SpeciesName*] = Tensorial ()

Scope

Aria Material

Summary

Generalized model for species diffusion - can use either a scalar or tensor species diffusivity. Identical to BASIC model.

Description

If a tensor species diffusivity D_i is provided in the material model, the form of flux is

$$\mathbf{J} = -D_i \cdot \nabla S_i$$

If the species diffusivity is given as a scalar d_i then the heat flux is

$$\mathbf{J} = -d_i \nabla S_i$$

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–

Tensorial_Dispersive

Syntax

Species Diffusion [*{of}* *SpeciesName*] = Tensorial_Dispersive (Alpha_T = *alpha_t* | Alpha_L = *alpha_l* | Tau = *tau* | [Epsilon = *epsilon*])

Scope

Aria Material

Description

The form of flux is

$$\mathbf{J} = -\left(\alpha_\mathbf{t}||\mathbf{v}|| + (\alpha_\mathbf{l} - \alpha_\mathbf{t})\frac{\mathbf{v}\times\mathbf{v}}{||\mathbf{v}||} + \tau\phi\mathbf{SD_i}\nabla\mathbf{C_i}\right)$$

where $\mathbf{v}$ is the velocity, τ the tortuosity, ϕ the porosity, S the saturation, D_i the species diffusivity and C_i the species concentration.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
alpha_t	real	–
alpha_l	real	–
tau	real	–
epsilon	real	1e-06

Thermophoresis

Syntax

Species Diffusion [*{of}* *SpeciesName*] = Thermophoresis ()

Scope

Aria Material

Description

The form of flux is

$$\mathbf{J} = -D_i \cdot \nabla C_i - S \nabla T$$

where D_i is the species diffusivity, C_i the species concentration, S the Soret coefficient, and T the temperature.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–

User_Field

Syntax

Species Diffusion [*{of}* *SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Vector from User Field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Species Diffusion [*{of}* *SpeciesName*] = User_Function ([Name_X = *name_x*] | [Name_Y = *name_y*] | [Name_Z = *name_z*] | X = *x* | [Multiplier = *multiplier*])

Scope

Aria Material

Summary

Value from a Vector User Function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name_x	"string"	–
name_y	"string"	–
name_z	"string"	–
x	"string"	–
multiplier	real	1

User_Plugin

Syntax

Species Diffusion [*{of}* *SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	“string”	–
plugin_parameters	“string”	–

Vector_String_Function

Syntax

Species Diffusion [*{of}* *SpeciesName*] = Vector_String_Function (F_X = *f_x* | F_Y = *f_y* | [F_Z = *f_z*])

Scope

Aria Material

Summary

Value from a string function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f_x	“string”	–
f_y	“string”	–
f_z	“string”	–

Species Diffusivity

Arrhenius

Syntax

Species Diffusivity [*{of}* *SpeciesName*] = Arrhenius (R = *r* | [C = *c*] | [D = *d*] | Q = *q*)

Scope

Aria Material

Summary

Arrhenius species diffusivity, $D_i = C + D * exp\left(-\frac{Q}{RT}\right)$

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
r	real	–
c	real	0
d	real	1
q	real	–

Constant

Syntax

Species Diffusivity [*{of} SpeciesName*] = Constant *{d | value}* = *d*

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
d	real	–

Copied

Syntax

Species Diffusivity [*{of} SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	“string”	–

Dissolution

Syntax

Species Diffusivity [*{of}* *SpeciesName*] = Dissolution D = *d*

Scope

Aria Material

Summary

Dissolution species diffusivity for a species dissolving in a liquid from a gas.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
d	real	–

Exponential

Syntax

Species Diffusivity [*{of}* *SpeciesName*] = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Scope

Aria Material

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Species Diffusivity [*{of}* *SpeciesName*] = Global (Global_Name = *global_name* | [Component = *component*])

Scope

Aria Material

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
global_name	“string”	–
component	integer	0

Nd_Table

Syntax

Species Diffusivity [*{of}* *SpeciesName*] = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Species Diffusivity [*{of}* *SpeciesName*] = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Species Diffusivity [*{of}* *SpeciesName*] = Scalar_String_Function F = *f*

Scope

Aria Material

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be

a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	"string"	–

User_Field

Syntax

Species Diffusivity [*{of}* *SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Species Diffusivity [*{of}* *SpeciesName*] = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Scope

Aria Material

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1
toggle	"string"	–

User_Plugin

Syntax

Species Diffusivity [*{of} SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	"string"	–
plugin_parameters	"string"	–

Species Expansion Coeff

Constant

Syntax

Species Expansion Coeff [*{of} SpeciesName*] = Constant *{species_expansion_coeff | value}* = *species_expansion_coeff*

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
species_expansion_coeff	real	–

Copied

Syntax

Species Expansion Coeff [*{of}* *SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	“string”	–

Exponential

Syntax

Species Expansion Coeff [*{of}* *SpeciesName*] = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Scope

Aria Material

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Species Expansion Coeff [*{of} SpeciesName*] = Global (Global_Name = *global_name* | [Component = *component*])

Scope

Aria Material

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Species Expansion Coeff [*{of} SpeciesName*] = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Species Expansion Coeff [*{of}* *SpeciesName*] = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Species Expansion Coeff [*{of} SpeciesName*] = Scalar_String_Function F = *f*

Scope

Aria Material

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	"string"	–

User_Field

Syntax

Species Expansion Coeff [*{of}* *SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Species Expansion Coeff [*{of}* *SpeciesName*] = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Scope

Aria Material

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Species Expansion Coeff [*{of} SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	“string”	–
plugin_parameters	“string”	–

Species Face Stabilization Scaling

Default

Syntax

Species Face Stabilization Scaling ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) = Default ()

Scope

Aria Material

Summary

Default face stabilization scaling for energy equation

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–

Species Fraction

Constant

Syntax

Species Fraction [*{of}* *SpeciesName*] = Constant Value = *value*

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
value	real	–

Copied

Syntax

Species Fraction [*{of}* *SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	“string”	–

Exponential

Syntax

Species Fraction [*{of}* *SpeciesName*] = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Scope

Aria Material

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

From_Chemeq

Syntax

Species Fraction [*{of}* *SpeciesName*] = From_Chemeq ()

Scope

Aria Material

Summary

Species fraction of one species extracted from ChemEq

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–

From_Mass_Fraction

Syntax

Species Fraction [*{of}* *SpeciesName*] = From_Mass_Fraction ()

Scope

Aria Material

Summary

Species fraction from mass fraction

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–

From_Species

Syntax

Species Fraction [*{of}* *SpeciesName*] = From_Species ()

Scope

Aria Material

Summary

Species fraction of one species from it's concentration and the overall concentration

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–

Global

Syntax

Species Fraction [*{of}* *SpeciesName*] = Global (Global_Name = *global_name* | [Component = *component*])

Scope

Aria Material

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
global_name	“string”	–
component	integer	0

Nd_Table

Syntax

Species Fraction [*{of}* *SpeciesName*] = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Species Fraction [*{of}* *SpeciesName*] = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Species Fraction [*{of}* *SpeciesName*] = Scalar_String_Function F = *f*

Scope

Aria Material

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	"string"	–

User_Field

Syntax

Species Fraction [*{of}* *SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Species Fraction [*{of}* *SpeciesName*] = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Scope

Aria Material

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1
toggle	"string"	–

User_Plugin

Syntax

Species Fraction [*{of}* *SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	"string"	–
plugin_parameters	"string"	–

Species Mobility

Constant

Syntax

Species Mobility [*{of}* *SpeciesName*] = Constant Value = *value*

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
value	real	–

Copied

Syntax

Species Mobility [*{of}* *SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	“string”	–

Exponential

Syntax

Species Mobility [*{of}* *SpeciesName*] = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Scope

Aria Material

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Species Mobility [*{of}* *SpeciesName*] = Global (Global_Name = *global_name* | [Component = *component*])

Scope

Aria Material

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Species Mobility [*{of}* *SpeciesName*] = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Nernst_Einstein

Syntax

Species Mobility [*{of}* *SpeciesName*] = Nernst_Einstein ([R = *r*] | [Temperature_Material_Phase = *temperature_material_phase*])

Scope

Aria Material

Summary

Mobility from diffusivity, M = D/(RT).

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
r	real	8.314
temperature_material_phase	“string”	–

Polynomial

Syntax

Species Mobility [*{of}* *SpeciesName*] = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Species Mobility [*{of}* *SpeciesName*] = Scalar_String_Function F = *f*

Scope

Aria Material

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	"string"	–

User_Field

Syntax

Species Mobility [*{of}* *SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Species Mobility [*{of}* *SpeciesName*] = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Scope

Aria Material

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Species Mobility [*{of}* *SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	“string”	–
plugin_parameters	“string”	–

Species Production

From_Time_Rate_Of_Change

Syntax

Species Production [*{of}* *SpeciesName*] = From_Time_Rate_Of_Change [Max_Order = *max_order*]

Scope

Aria Material

Summary

Species production rate from the time derivative of species concentration. This is done by default using a first-order time integration scheme (new - old)/dt To use the same order as the overall solve, set max_order > 1

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
max_order	integer	1

Species Surface

Constant

Syntax

Species Surface [*{of}* *SpeciesName*] = Constant Value = *value*

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
value	real	–

Copied

Syntax

Species Surface [*{of}* *SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	“string”	–

Diffusion_To_Surf_Correction

Syntax

Species Surface [*{of} SpeciesName*] = Diffusion_To_Surf_Correction (Species_A = *species_a* | Species_B = *species_b* | D = *d* | R = *r* | F = *f* | K = *k* | Eqpot = *eqPot* | Dir = *dir*)

Scope

Aria Material

Summary

Correction for surface concentration based on balancing reaction rate and diffusion from pore center to wall.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
species_a	"string"	–
species_b	"string"	–
d	real	–
r	real	–
f	real	–
k	real	–
eqPot	real	–
dir	real	–

Equivalent_To_Bulk

Syntax

Species Surface [*{of} SpeciesName*] = Equivalent_To_Bulk ()

Scope

Aria Material

Summary

Approximate a surface species concentration as identical to the bulk concentration

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–

Exponential

Syntax

Species Surface [*{of}* *SpeciesName*] = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Scope

Aria Material

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Species Surface [*{of}* *SpeciesName*] = Global (Global_Name = *global_name* | [Component = *component*])

Scope

Aria Material

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Species Surface [*{of}* *SpeciesName*] = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Species Surface [*{of}* *SpeciesName*] = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Species Surface [*{of} SpeciesName*] = Scalar_String_Function F = *f*

Scope

Aria Material

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	"string"	–

User_Field

Syntax

Species Surface [*{of}* *SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Species Surface [*{of}* *SpeciesName*] = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Scope

Aria Material

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Species Surface [*{of} SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	“string”	–
plugin_parameters	“string”	–

Species Valence

Constant

Syntax

Species Valence [*{of} SpeciesName*] = Constant *{value | z}* = *z*

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
z	real	–

Copied

Syntax

Species Valence [*{of}* *SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	“string”	–

Exponential

Syntax

Species Valence [*{of}* *SpeciesName*] = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Scope

Aria Material

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Species Valence [*{of} SpeciesName*] = Global (Global_Name = *global_name* | [Component = *component*])

Scope

Aria Material

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Species Valence [*{of} SpeciesName*] = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Species Valence [*{of}* *SpeciesName*] = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Species Valence [*{of}* *SpeciesName*] = Scalar_String_Function F = *f*

Scope

Aria Material

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	"string"	–

User_Field

Syntax

Species Valence [*{of}* *SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Species Valence [*{of}* *SpeciesName*] = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Scope

Aria Material

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Species Valence [*{of}* *SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	“string”	–
plugin_parameters	“string”	–

Specific Dissipation Rate

Constant

Syntax

Specific Dissipation Rate [*{of}* *SpeciesName*] = Constant Value = *value*

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
value	real	–

Copied

Syntax

Specific Dissipation Rate [*{of}* *SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	“string”	–

Exponential

Syntax

Specific Dissipation Rate [*{of}* *SpeciesName*] = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Scope

Aria Material

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Specific Dissipation Rate [*{of} SpeciesName*] = Global (Global_Name = *global_name* | [Component = *component*])

Scope

Aria Material

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Specific Dissipation Rate [*{of} SpeciesName*] = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Specific Dissipation Rate [*{of} SpeciesName*] = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Specific Dissipation Rate [*{of} SpeciesName*] = Scalar_String_Function F = *f*

Scope

Aria Material

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	"string"	–

User_Field

Syntax

Specific Dissipation Rate [*{of}* *SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Specific Dissipation Rate [*{of}* *SpeciesName*] = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Scope

Aria Material

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Specific Dissipation Rate [*{of} SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	“string”	–
plugin_parameters	“string”	–

Specific Dissipation Rate Density

Constant

Syntax

Specific Dissipation Rate Density [*{of} SpeciesName*] = Constant Value = *value*

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
value	real	–

Copied

Syntax

Specific Dissipation Rate Density [*{of}* *SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	“string”	–

Exponential

Syntax

Specific Dissipation Rate Density [*{of}* *SpeciesName*] = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Scope

Aria Material

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Specific Dissipation Rate Density [*{of}* *SpeciesName*] = Global (Global_Name = *global_name* | [Component = *component*])

Scope

Aria Material

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Specific Dissipation Rate Density [*{of}* *SpeciesName*] = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Specific Dissipation Rate Density [*{of}* *SpeciesName*] = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Specific Dissipation Rate Density [*{of}* *SpeciesName*] = Scalar_String_Function F = *f*

Scope

Aria Material

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	"string"	–

User_Field

Syntax

Specific Dissipation Rate Density [*{of}* *SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Specific Dissipation Rate Density [*{of}* *SpeciesName*] = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Scope

Aria Material

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Specific Dissipation Rate Density [*{of}* *SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	“string”	–
plugin_parameters	“string”	–

Specific Dissipation Rate Diffusive Flux

Basic

Syntax

Specific Dissipation Rate Diffusive Flux [*{of}* *SpeciesName*] = Basic ()

Scope

Aria Material

Summary

Basic model of specific dissipation rate diffusive flux

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–

Specific Heat

Calore_User_Sub

Syntax

Specific Heat [*{of} SpeciesName*] = Calore_User_Sub (Name = *name* | Type = *type* | [Multiplier = *multiplier*] | [Material_Data_Block = *material_data_block*] | [Data = *data*] | [Scaling_Field = *scaling_field*])

Scope

Aria Material

Summary

Values from a Calore user subroutine

Description

No enthalpy evaluator is provided.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
type	"string"	–
multiplier	real	1
material_data_block	"string"	–
data	"string"	–
scaling_field	"string"	–

Cantera

Syntax

Specific Heat [*{of} SpeciesName*] = Cantera [Poffset = *poffset*]

Scope

Aria Material

Summary

Evaluates the specific heat using the Cantera library.

Description

A valid Cantera XML File must be provided elsewhere in the material definition.

No enthalpy evaluator is provided.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
poffset	real	0

Constant

Syntax

Specific Heat [*{of}* *SpeciesName*] = Constant ([H0 = *h0*] | [S0 = *s0*] | [T_Standard = *T_standard*] | [P_Standard = *P_standard*] | [Cp = *Cp*] | [Value = *value*])

Scope

Aria Material

Summary

Constant specific heat from an evaluator

Description

An enthalpy evaluator is automatically provided, the reference state temperature, enthalpy, and entropy can optionally be provided via the T_standard, h0, and s0 parameters. All default to 0 if they are not provided.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
h0	real	0
s0	real	0
T_standard	real	0
P_standard	real	0
Cp	real	0
value	real	0

Curing_Foam

Syntax

Specific Heat [*{of}* *SpeciesName*] = Curing_Foam ()

Scope

Aria Material

Summary

Specific heat model for a curing polymer foam.

Description

$$c_p = \frac{c_{p,g}\rho_g\phi_g + c_{p,l}\rho_l(1-\phi_g)}{\rho}$$

where ρ is overall density, ρ_g is the gas density, ϕ_g is the volume fraction gas, and ρ_l is the liquid density.

No enthalpy evaluator is provided.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–

Exponential

Syntax

Specific Heat [*{of}* *SpeciesName*] = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Scope

Aria Material

Summary

Value from an exponential function

Description

$$c_p = C + Me^{kX}$$

where X is the variable, C is the constant, M is the multiplier, and k is exponent. C and M are optional and default to 0 and 1 respectively.

No enthalpy evaluator is provided when using an exponential specific heat.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	“string”	–
constant	real	0
multiplier	real	1
exponent	real	–

Fortran

Syntax

Specific Heat [*{of}* *SpeciesName*] = Fortran ([Multiplier = *multiplier*] | Sub_Name = *sub_name* | [Real_Data = *real_data*] | [Int_Data = *int_data*] | [Resource_Name = *resource_name*] | [Data = *data*])

Scope

Aria Material

Summary

Fortran subroutine specific heat.

Description

No enthalpy evaluator is provided.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
multiplier	real	1
sub_name	“string”	–
real_data	“string”	–
int_data	“string”	–
resource_name	“string”	–
data	“string”	–

Global

Syntax

Specific Heat [*{of}* *SpeciesName*] = Global (Global_Name = *global_name* | [Component = *component*])

Scope

Aria Material

Summary

Value from a global variable

Description

No enthalpy evaluator is provided when using a global variable specific heat.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
global_name	"string"	–
component	integer	0

Interpolated_Phase_Average

Syntax

Specific Heat [*{of}* *SpeciesName*] = Interpolated_Phase_Average ([Subindex_A = *subindex_a*] | [Subindex_B = *subindex_b*])

Scope

Aria Material

Summary

Interpolated level set phase averaged specific heat.

Description

See *Phase_Average*, however in this model the evaluation is done based on nodal values and interpolated to the integration points.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
subindex_a	integer	-1
subindex_b	integer	-1

Mass_Average

Syntax

Specific Heat [*{of}* *SpeciesName*] = Mass_Average ()

Scope

Aria Material

Summary

Normalized mass average of species specific heats.

Description

$$c_p = \frac{\sum_{i \in \text{species}} \left[\text{clip}(Y_i) c_{p,i}\right]}{\sum_{i \in \text{species}} \text{clip}(Y_i)}$$

where Y_i is the mass fraction of species i, $c_{p,i}$ the specific heat of species i and the clip() function clips values to the bounds [0, 1].

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–

Nasa14

Syntax

Specific Heat [*{of}* *SpeciesName*] = Nasa14 ([H0 = *h0*] | [S0 = *s0*] | [T_Standard = *T_standard*] | [P_Standard = *P_standard*] | Tmid = *Tmid* | Tmax = *Tmax* | R = *r* | [L0 = *l0*] | [L1 = *l1*] | [L2 = *l2*] | [L3 = *l3*] | [L4 = *l4*] | [L5 = *l5*] | [L6 = *l6*] | [L = *l1 l2 l3 l4 l5 l6 l7*] | [H1 = *h1*] | [H2 = *h2*] | [H3 = *h3*] | [H4 = *h4*] | [H5 = *h5*] | [H6 = *h6*] | [H = *h1 h2 h3 h4 h5 h6 h7*] | [T_Offset = *t_offset*])

Scope

Aria Material

Summary

Value from a NASA14 polynomial function

Description

$$T_i = T + T_{\text{offset}}$$

$$c_p = \begin{cases} R(L_0 + L_1 T_i + L_2 T_i^2 + L_3 T_i^3 + L_4 T_i^4), T_i <= T_{\text{mid}} \\ R(H_0 + H_1 T_i + H_2 T_i^2 + H_3 T_i^3 + H_4 T_i^4), T_i > T_{\text{mid}} \text{ and } T <= T_{\text{max}} \\ R(H_0 + H_1 T_{\text{max}} + H_2 T_{\text{max}}^2 + H_3 T_{\text{max}}^3 + H_4 T_{\text{max}}^4), T_i > T_{\text{max}} \end{cases}$$

An enthalpy evaluator is automatically provided, the reference state enthalpy, and entropy can be adjusted with the h0 and s0 parameters which are added to the appropriate NASA polynomial coefficients.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
h0	real	0
s0	real	0
T_standard	real	0
P_standard	real	0
Tmid	real	–
Tmax	real	–
r	real	–
l0	real	0
l1	real	0
l2	real	0
l3	real	0
l4	real	0
l5	real	0
l6	real	0
l	real1 real2 real3 real4 real5 real6 real7	0
h1	real	0
h2	real	0
h3	real	0
h4	real	0
h5	real	0
h6	real	0
h	real1 real2 real3 real4 real5 real6 real7	0
t_offset	real	0

Nd_Table

Syntax

Specific Heat [*{of} SpeciesName*] = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Phase_Average

Syntax

Specific Heat [*{of}* *SpeciesName*] = Phase_Average ([Subindex_A = *subindex_a*] | [Subindex_B = *subindex_b*])

Scope

Aria Material

Summary

Level set phase averaged specific heat.

Description

$$c_p = (1 - H(\phi))c_{p,A} + H(\phi)c_{p,B}$$

where $H(\phi)$ is the heaviside function evaluated at the level-set variable ϕ, $c_{p,A}$ is the specific heat in phase A, and $c_{p,B}$ is the specific heat in phase B.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
subindex_a	integer	-1
subindex_b	integer	-1

Polynomial

Syntax

Specific Heat [*{of}* *SpeciesName*] = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*] | [H0 = *h0*] | [S0 = *s0*] | [T_Standard = *T_standard*] | [P_Standard = *P_standard*] | [R = *r*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

$$c_p = R(C_0 + \sum_{\text{n=1}}^{\text{order}} C_i(T + T_{off})^n)$$

where T_{off} is set by the "VARIABLE_OFFSET" parameter.

An enthalpy evaluator is automatically provided, the reference state temperature, enthalpy, and entropy can optionally be provided via the T_standard, h0, and s0 parameters. All default to 0 if they are not provided.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0
h0	real	0
s0	real	0
T_standard	real	0
P_standard	real	0
r	real	1

Porous_Phase_Specific_Average

Syntax

Specific Heat [*{of} SpeciesName*] = Porous_Phase_Specific_Average ([Liquid_Phase_Name = *Liquid_Phase_Name*] | [Gas_Phase_Name = *Gas_Phase_Name*])

Scope

Aria Material

Summary

Specific heat for a multiphase material of two or three phases

Description

For a two-phase (solid and liquid) porous media problem:

$$c_p = \frac{(1-\phi)\rho_s c_{p,s} + \phi\rho_l c_{p,l}}{\rho}$$

For a three-phase (solid, liquid, gas) porous media problem:

$$c_p = \frac{(1-\phi)\rho_s c_{p,s} + \phi S_l \rho_l c_{p,l} + \phi(1-S_l)\rho_g c_{p,g}}{\rho}$$

where ϕ is the porosity, ρ_i the density in phase i, $c_{p,i}$ the specific heat in phase i, and S_l the liquid saturation.

No enthalpy evaluator is provided.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Liquid_Phase_Name	“string”	–
Gas_Phase_Name	“string”	–

Scalar_String_Function

Syntax

Specific Heat [*{of} SpeciesName*] = Scalar_String_Function ([H0 = *h0*] | [S0 = *s0*] | [T_Standard = *T_standard*] | [P_Standard = *P_standard*] | [T_0 = *t_0*] | [T_F = *T_f*] | [Dt = *dT*] | F = *f*)

Scope

Aria Material

Summary

Compute specific heat from a string function.

Description

If the only variable in the string function is temperature then this will build a piecewise linear table of Temperature vs. c_p based on this string function to use for an enthalpy evaluator. The default range of temperature to evaluate is T_0 = 100, T_f = 5000, and dT = 10. These can be overridden as optional parameters.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
h0	real	0
s0	real	0
T_standard	real	0
P_standard	real	0
t_0	real	100
T_f	real	5000
dT	real	10
f	"string"	–

T_Exponent

Syntax

Specific Heat [*{of} SpeciesName*] = T_Exponent ([H0 = *h0*] | [S0 = *s0*] | [T_Standard = *T_standard*] | [P_Standard = *P_standard*] | Cp_Ref = *Cp_ref* | T_Ref = *T_ref* | N = *n*)

Scope

Aria Material

Summary

Temperature exponent specific heat from an evaluator

Description

$$c_p = c_{p,\text{ref}} \left(\frac{T}{T_{\text{ref}}} \right)^n$$

An enthalpy evaluator is automatically provided, the reference state temperature, enthalpy, and entropy can optionally be provided via the T_standard, h0, and s0 parameters. All default to 0 if they are not provided.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
h0	real	0
s0	real	0
T_standard	real	0
P_standard	real	0
Cp_ref	real	–
T_ref	real	–
n	real	–

Use_Phase_Change

Syntax

Specific Heat [*{of}* *SpeciesName*] = Use_Phase_Change (Flh = *flh* | Ts = *ts* | Tl = *tl*)

Scope

Aria Material

Summary

For versions > 4.53 this material model automatically creates a "MELTING" source term for the energy equation and uses the accompanying PHASE CHANGE SPECIFIC HEAT model to define the specific heat.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
flh	real	–
ts	real	–
tl	real	–

User_Field

Syntax

Specific Heat [*{of}* *SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Value from a user field

Description

No enthalpy evaluator is provided when using a user field for specific heat.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Specific Heat [*{of} SpeciesName*] = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*] | [H0 = *h0*] | [S0 = *s0*] | [T_Standard = *T_standard*] | [P_Standard = *P_standard*])

Scope

Aria Material

Summary

Value from a user function

Description

If X is temperature then an enthalpy evaluator is automatically provided. The reference state temperature, enthalpy, and entropy can optionally be provided via the T_standard, h0, and s0 parameters. All default to 0 if they are not provided.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1
toggle	"string"	–
h0	real	0
s0	real	0
T_standard	real	0
P_standard	real	0

User_Plugin

Syntax

Specific Heat [*{of}* *SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	"string"	–
plugin_parameters	"string"	–

Specific Heat Cp

Cantera

Syntax

Specific Heat Cp [*{of}* *SpeciesName*] = Cantera [Poffset = *poffset*]

Scope

Aria Material

Summary

Cantera specific heat for low Mach number applications

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
poffset	real	0

Constant

Syntax

Specific Heat Cp [*{of}* *SpeciesName*] = Constant Value = *value*

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
value	real	–

Copied

Syntax

Specific Heat Cp [*{of}* *SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	“string”	–

Exponential

Syntax

Specific Heat Cp [*{of}* *SpeciesName*] = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Scope

Aria Material

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Specific Heat Cp [*{of}* *SpeciesName*] = Global (Global_Name = *global_name* | [Component = *component*])

Scope

Aria Material

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Specific Heat Cp [*{of}* *SpeciesName*] = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Specific Heat Cp [*{of}* *SpeciesName*] = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Specific Heat Cp [*{of} SpeciesName*] = Scalar_String_Function F = *f*

Scope

Aria Material

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	"string"	–

User_Field

Syntax

Specific Heat Cp [*{of} SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Specific Heat Cp [*{of} SpeciesName*] = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Scope

Aria Material

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Specific Heat Cp [*{of}* *SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	“string”	–
plugin_parameters	“string”	–

Specific Surface Area

Constant

Syntax

Specific Surface Area [*{of}* *SpeciesName*] = Constant Value = *value*

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
value	real	–

Copied

Syntax

Specific Surface Area [*{of}* *SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	“string”	–

Exponential

Syntax

Specific Surface Area [*{of}* *SpeciesName*] = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Scope

Aria Material

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Specific Surface Area [*{of}* *SpeciesName*] = Global (Global_Name = *global_name* | [Component = *component*])

Scope

Aria Material

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
global_name	"string"	–
component	integer	0

Mmp_Model

Syntax

Specific Surface Area [*{of}* *SpeciesName*] = Mmp_Model Rho_C_Init = *rho_c_init*

Scope

Aria Material

Summary

Computes the specific surface area based on the micromechanics pressurization model

Description

Computes the specific surface area (SSA) pore damage metric via the MMP model. This is computed as $SSA = \frac{3/(r)}{\rho_{pg}+\left(\frac{\rho_{co}}{1+\varepsilon_T}\right)\left(\phi^{-1}-1\right)}$. Here, r is the PORE DEFECT RADIUS, ρ_{pg} is the PORE GAS DENSITY, ρ_{co} is the initial

condensed density, ε_T is the thermal strain (computed from the PORE DEFECT RADIUS model), and ϕ is the gas volume fraction.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
rho_c_init	real	–

Nd_Table

Syntax

Specific Surface Area [*{of} SpeciesName*] = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Specific Surface Area [*{of}* *SpeciesName*] = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Specific Surface Area [*{of}* *SpeciesName*] = Scalar_String_Function F = *f*

Scope

Aria Material

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be

a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	"string"	–

User_Field

Syntax

Specific Surface Area [*{of}* *SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Specific Surface Area [*{of}* *SpeciesName*] = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Scope

Aria Material

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1
toggle	"string"	–

User_Plugin

Syntax

Specific Surface Area [*{of} SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	"string"	–
plugin_parameters	"string"	–

Supg Tau

Shakib_Charge_Density

Syntax

Supg Tau ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) = Shakib_Charge_Density ([H = *h*] | [Use_Advection = *use_advection*] | [Use_Diffusion = *use_diffusion*] | [Use_Time = *use_time*] | [Multiplier = *multiplier*])

Scope

Aria Material

Summary

A Shakib SUPG Tau model

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
h	real	0
use_advection	integer	1
use_diffusion	integer	1
use_time	integer	0
multiplier	real	1

Shakib_Cvfem_Dispersed_Phase_Momentum

Syntax

Supg Tau ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) = Shakib_Cvfem_Dispersed_Phase_Momentum ([Multiplier = *multiplier*] | [Use_Diffusion = *use_diffusion*] | [Use_Time = *use_time*])

Scope

Aria Material

Summary

A Shakib SUPG Tau model

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
multiplier	real	1
use_diffusion	integer	–
use_time	integer	–

Shakib_Cvfem_Level_Set

Syntax

Supg Tau ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) = Shakib_Cvfem_Level_Set ([Multiplier = *multiplier*] | [Use_Diffusion = *use_diffusion*] | [Use_Time = *use_time*])

Scope

Aria Material

Summary

A Shakib SUPG Tau model

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
multiplier	real	1
use_diffusion	integer	–
use_time	integer	–

Shakib_Cvfem_Mixture_Fraction

Syntax

Supg Tau ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) = Shakib_Cvfem_Mixture_Fraction ([Multiplier = *multiplier*] | [Use_Diffusion = *use_diffusion*] | [Use_Time = *use_time*])

Scope

Aria Material

Summary

A Shakib SUPG Tau model

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
multiplier	real	1
use_diffusion	integer	–
use_time	integer	–

Shakib_Cvfem_Momentum

Syntax

Supg Tau ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) = Shakib_Cvfem_Momentum ([Multiplier = *multiplier*] | [Use_Diffusion = *use_diffusion*] | [Use_Time = *use_time*])

Scope

Aria Material

Summary

A Shakib SUPG Tau model

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
multiplier	real	1
use_diffusion	integer	–
use_time	integer	–

Shakib_Energy

Syntax

Supg Tau ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) = Shakib_Energy ([H = *h*] | [Use_Advection = *use_advection*] | [Use_Diffusion = *use_diffusion*] | [Use_Time = *use_time*] | [Multiplier = *multiplier*])

Scope

Aria Material

Summary

A Shakib SUPG Tau model

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
h	real	0
use_advection	integer	1
use_diffusion	integer	1
use_time	integer	0
multiplier	real	1

Shakib_Enthalpy

Syntax

Supg Tau ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) = Shakib_Enthalpy ([H = *h*] | [Use_Advection = *use_advection*] | [Use_Diffusion = *use_diffusion*] | [Use_Time = *use_time*] | [Multiplier = *multiplier*])

Scope

Aria Material

Summary

A Shakib SUPG Tau model

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
h	real	0
use_advection	integer	1
use_diffusion	integer	1
use_time	integer	0
multiplier	real	1

Shakib_Level_Set

Syntax

Supg Tau ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) = Shakib_Level_Set ([H = *h*] | [Use_Advection = *use_advection*] | [Use_Diffusion = *use_diffusion*] | [Use_Time = *use_time*] | [Multiplier = *multiplier*])

Scope

Aria Material

Summary

A Shakib SUPG Tau model

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
h	real	0
use_advection	integer	1
use_diffusion	integer	1
use_time	integer	0
multiplier	real	1

Shakib_Mass_Balance

Syntax

Supg Tau ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) = Shakib_Mass_Balance ([H = *h*] | [Use_Advection = *use_advection*] | [Use_Diffusion = *use_diffusion*] | [Use_Time = *use_time*] | [Multiplier = *multiplier*])

Scope

Aria Material

Summary

A Shakib SUPG Tau model

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
h	real	0
use_advection	integer	1
use_diffusion	integer	1
use_time	integer	0
multiplier	real	1

Shakib_Mixture_Fraction

Syntax

Supg Tau ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) = Shakib_Mixture_Fraction ([H = *h*] | [Use_Advection = *use_advection*] | [Use_Diffusion = *use_diffusion*] | [Use_Time = *use_time*] | [Multiplier = *multiplier*])

Scope

Aria Material

Summary

A Shakib SUPG Tau model

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
h	real	0
use_advection	integer	1
use_diffusion	integer	1
use_time	integer	0
multiplier	real	1

Shakib_Momentum

Syntax

Supg Tau ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) = Shakib_Momentum ([H = *h*] | [Use_Advection = *use_advection*] | [Use_Diffusion = *use_diffusion*] | [Use_Time = *use_time*] | [Multiplier = *multiplier*])

Scope

Aria Material

Summary

A Shakib SUPG Tau model

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
h	real	0
use_advection	integer	1
use_diffusion	integer	1
use_time	integer	0
multiplier	real	1

Shakib_Species

Syntax

Supg Tau ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) = Shakib_Species ([H = *h*] | [Use_Advection = *use_advection*] | [Use_Diffusion = *use_diffusion*] | [Use_Time = *use_time*] | [Multiplier = *multiplier*])

Scope

Aria Material

Summary

A Shakib SUPG Tau model

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
h	real	0
use_advection	integer	1
use_diffusion	integer	1
use_time	integer	0
multiplier	real	1

Shakib_Suspension

Syntax

Supg Tau ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) = Shakib_Suspension ([H = *h*] | [Use_Advection = *use_advection*] | [Use_Diffusion = *use_diffusion*] | [Use_Time = *use_time*] | [Multiplier = *multiplier*])

Scope

Aria Material

Summary

A Shakib SUPG Tau model for the suspension equation

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
h	real	0
use_advection	integer	1
use_diffusion	integer	1
use_time	integer	0
multiplier	real	1

Surface Tension

Constant

Syntax

Surface Tension [*{of} SpeciesName*] = Constant *{sigma | value}* = *sigma*

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
sigma	real	–

Copied

Syntax

Surface Tension [*{of} SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	“string”	–

Exponential

Syntax

Surface Tension [*{of} SpeciesName*] = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Scope

Aria Material

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Surface Tension [*{of} SpeciesName*] = Global (Global_Name = *global_name* | [Component = *component*])

Scope

Aria Material

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
global_name	"string"	–
component	integer	0

Linear_T

Syntax

Surface Tension [*{of} SpeciesName*] = Linear_T (Sigma0 = *sigma0* | Dsigmadt = *dsigmadT* | T_Ref = *T_ref*)

Scope

Aria Material

Summary

Linear T surface tension

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
sigma0	real	–
dsigmadT	real	–
T_ref	real	–

Nd_Table

Syntax

Surface Tension [*{of}* *SpeciesName*] = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Surface Tension [*{of}* *SpeciesName*] = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Surface Tension [*{of}* *SpeciesName*] = Scalar_String_Function F = *f*

Scope

Aria Material

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be

a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	"string"	–

User_Field

Syntax

Surface Tension [*{of}* *SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Surface Tension [*{of}* *SpeciesName*] = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Scope

Aria Material

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1
toggle	"string"	–

User_Plugin

Syntax

Surface Tension [*{of}* *SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	"string"	–
plugin_parameters	"string"	–

Suspension Flux

Balance

Syntax

Suspension Flux [*{of}* *SpeciesName*] = Balance (A = *a* | Mu_S = *mu_s* | Lambda_1 = *lambda_1* | Lambda_2 = *lambda_2* | Lambda_3 = *lambda_3*)

Scope

Aria Material

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
a	real	–
mu_s	real	–
lambda_1	real	–
lambda_2	real	–
lambda_3	real	–

Fad_Phillips

Syntax

Suspension Flux [*{of}* *SpeciesName*] = Fad_Phillips (K_Mu = *k_mu* | K_C = *k_c* | Phi_Max = *phi_max* | Beta = *beta* | Particle_Radius = *particle_radius* | [K_0 = *k_0*] | [Phi_Tol = *phi_tol*])

Scope

Aria Material

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
k_mu	real	–
k_c	real	–
phi_max	real	–
beta	real	–
particle_radius	real	–
k_0	real	0
phi_tol	real	0.0001

Hydrostatic

Syntax

Suspension Flux [*{of}* *SpeciesName*] = Hydrostatic (Gx = *gx* | Gy = *gy* | [Gz = *gz*] | Lambda = *lambda* | Fluid_Density = *fluid_density* | B = *b* | Phi_Max = *phi_max*)

Scope

Aria Material

Summary

This model has been deprecated

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
gx	real	–
gy	real	–
gz	real	0
lambda	real	–
fluid_density	real	–
b	real	–
phi_max	real	–

Phillips

Syntax

Suspension Flux [*{of}* *SpeciesName*] = Phillips (K_Mu = *k_mu* | K_C = *k_c* | Phi_Max = *phi_max* | Beta = *beta* | Particle_Radius = *particle_radius* | [K_0 = *k_0*] | [Phi_Tol = *phi_tol*] | [Use_Fd = *use_fd*])

Scope

Aria Material

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
k_mu	real	–
k_c	real	–
phi_max	real	–
beta	real	–
particle_radius	real	–
k_0	real	0
phi_tol	real	0.0001
use_fd	integer	–

Suspension Hindrance Function

Morris

Syntax

Suspension Hindrance Function [*{of}* *SpeciesName*] = Morris (Alpha = *alpha* | Phi_Max = *phi_max* | Phi_Tol = *phi_tol*)

Scope

Aria Material

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
alpha	real	–
phi_max	real	–
phi_tol	real	–

Suspension Normal Viscosity

Morris

Syntax

Suspension Normal Viscosity [*{of}* *SpeciesName*] = Morris ([K_N = *K_n*] | Phi_Max = *phi_max* | Phi_Tol = *phi_tol* | Mu_S = *mu_s*)

Scope

Aria Material

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
K_n	real	0.75
phi_max	real	–
phi_tol	real	–
mu_s	real	–

Suspension Q Tensor

Ct_Aligned

Syntax

Suspension Q Tensor [*{of}* *SpeciesName*] = Ct_Aligned (Lambda_1 = *lambda_1* | Lambda_2 = *lambda_2* | Lambda_3 = *lambda_3*)

Scope

Aria Material

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
lambda_1	real	–
lambda_2	real	–
lambda_3	real	–

Flow_Aligned

Syntax

Suspension Q Tensor [*{of}* *SpeciesName*] = *Flow_Aligned* (Lambda_A = *lambda_a* | Lambda_B = *lambda_b*)

Scope

Aria Material

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
lambda_a	real	–
lambda_b	real	–

Isotropic

Syntax

Suspension Q Tensor [*{of}* *SpeciesName*] = Isotropic ()

Scope

Aria Material

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–

T

Temperature

Calore_User_Sub

Syntax

Temperature [*{of} SpeciesName*] = Calore_User_Sub ([Power_Output = *power_output*] | [Flux_Output = *flux_output*] | [Toggle = *toggle*] | Name = *name* | Type = *type* | [Multiplier = *multiplier*] | [Material_Data_Block = *material_data_block*] | [Data = *data*])

Scope

Aria Material

Summary

Values from a Calore user subroutine

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output

If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Flux_Output

If specified, the average flux of the given BC/Src is calculated and stored into the global variable name specified as the *flux_output* argument.

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
power_output	“string”	–
flux_output	“string”	–
toggle	“string”	–
name	“string”	–
type	“string”	–
multiplier	real	1
material_data_block	“string”	–
data	“string”	–

Cantera

Syntax

Temperature [*{of}* *SpeciesName*] = Cantera ([Power_Output = *power_output*] | [Flux_Output = *flux_output*] | [Toggle = *toggle*] | [Omit_Fd_Sens = *omit_fd_sens*])

Scope

Aria Material

Summary

Cantera enthalpy computed from temperature

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output

If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Flux_Output

If specified, the average flux of the given BC/Src is calculated and stored into the global variable name specified as the *flux_output* argument.

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Omit_Fd_Sens

Omits calculating this sensitivity when using FD sensitivities when `true`.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
power_output	“string”	–
flux_output	“string”	–
toggle	“string”	–
omit_fd_sens	“string”	false

Cht_Robin

Syntax

Temperature [*{of}* *SpeciesName*] = Cht_Robin ([Power_Output = *power_output*] | [Flux_Output = *flux_output*] | [Toggle = *toggle*] | Temperature_Field = *temperature_field*)

Scope

Aria Material

Summary

Dirichlet boundary condition for interface coupled with Robin style BC CHT_ROBIN. This BC computes the heat flux for the other side.

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output

If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Flux_Output

If specified, the average flux of the given BC/Src is calculated and stored into the global variable name specified as the *flux_output* argument.

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
power_output	"string"	–
flux_output	"string"	–
toggle	"string"	–
temperature_field	"string"	–

Constant

Syntax

Temperature [*{of}* *SpeciesName*] = Constant ([Power_Output = *power_output*] | [Flux_Output = *flux_output*] | [Toggle = *toggle*] | Value = *value*)

Scope

Aria Material

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output

If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Flux_Output

If specified, the average flux of the given BC/Src is calculated and stored into the global variable name specified as the *flux_output* argument.

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
power_output	"string"	–
flux_output	"string"	–
toggle	"string"	–
value	real	–

Copied

Syntax

Temperature [*{of} SpeciesName*] = Copied ([Power_Output = *power_output*] | [Flux_Output = *flux_output*] | [Toggle = *toggle*] | Source = *source*)

Scope

Aria Material

Summary

Copied value from another expression

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output

If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Flux_Output

If specified, the average flux of the given BC/Src is calculated and stored into the global variable name specified as the *flux_output* argument.

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
power_output	"string"	–
flux_output	"string"	–
toggle	"string"	–
source	"string"	–

Correlation_Fluid

Syntax

Temperature [*{of} SpeciesName*] = Correlation_Fluid ([T = *t*] | [Bulk_Node_Interface = *bulk_node_interface*])

Scope

Aria Material

Summary

Correlation fluid temperature

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
t	real	0
bulk_node_interface	"string"	–

Correlation_Wall

Syntax

Temperature [*{of}* *SpeciesName*] = Correlation_Wall ([Tw = *Tw*] | [Bulk_Node_Interface = *bulk_node_interface*])

Scope

Aria Material

Summary

Correlation wall temperature

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Tw	real	0
bulk_node_interface	"string"	–

Exponential

Syntax

Temperature [*{of}* *SpeciesName*] = Exponential ([Power_Output = *power_output*] | [Flux_Output = *flux_output*] | [Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Scope

Aria Material

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the

constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output
If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Flux_Output
If specified, the average flux of the given BC/Src is calculated and stored into the global variable name specified as the *flux_output* argument.

Toggle
If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
power_output	"string"	–
flux_output	"string"	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Fortran_User_Sub

Syntax
Temperature [*{of}* *SpeciesName*] = Fortran_User_Sub ([Power_Output = *power_output*] | [Flux_Output = *flux_output*] | [Toggle = *toggle*] | [Multiplier = *multiplier*] | Name = *name* | Type = *type* | [Real_Data = *real_data*] | [Int_Data = *int_data*] | [Resource_Name = *resource_name*] | [Data = *data*] | [Material_Data_Block = *material_data_block*])

Scope
Aria Material

Summary
Dirichlet value from a Fortran user subroutine

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output

If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Flux_Output

If specified, the average flux of the given BC/Src is calculated and stored into the global variable name specified as the *flux_output* argument.

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
power_output	"string"	–
flux_output	"string"	–
toggle	"string"	–
multiplier	real	1
name	"string"	–
type	"string"	–
real_data	"string"	–
int_data	"string"	–
resource_name	"string"	–
data	"string"	–
material_data_block	"string"	–

From_No_Material_Phase

Syntax

Temperature [*{of} SpeciesName*] = From_No_Material_Phase ([Power_Output = *power_output*] | [Flux_Output = *flux_output*] | [Toggle = *toggle*])

Scope

Aria Material

Summary

Copy the temperature from NO_MATERIAL_PHASE to the desired material phase.

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output

If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Flux_Output

If specified, the average flux of the given BC/Src is calculated and stored into the global variable name specified as the *flux_output* argument.

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
power_output	"string"	–
flux_output	"string"	–
toggle	"string"	–

Global

Syntax

Temperature [*{of}* *SpeciesName*] = Global ([Power_Output = *power_output*] | [Flux_Output = *flux_output*] | [Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Scope

Aria Material

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output

If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Flux_Output

If specified, the average flux of the given BC/Src is calculated and stored into the global variable name specified as the *flux_output* argument.

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
power_output	"string"	–
flux_output	"string"	–
toggle	"string"	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Temperature [*{of} SpeciesName*] = Nd_Table ([Power_Output = *power_output*] | [Flux_Output = *flux_output*] | [Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output

If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Flux_Output

If specified, the average flux of the given BC/Src is calculated and

stored into the global variable name specified as the *flux_output* argument.

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
power_output	“string”	–
flux_output	“string”	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Temperature [*{of}* *SpeciesName*] = Polynomial ([Power_Output = *power_output*] | [Flux_Output = *flux_output*] | [Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output

If specified, the total (integrated) power of the given BC/Src is

calculated and stored into the global variable name specified as the *power_output* argument.

Flux_Output

If specified, the average flux of the given BC/Src is calculated and stored into the global variable name specified as the *flux_output* argument.

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
power_output	"string"	–
flux_output	"string"	–
toggle	"string"	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Temperature [*{of}* *SpeciesName*] = Scalar_String_Function ([Power_Output = *power_output*] | [Flux_Output = *flux_output*] | [Toggle = *toggle*] | F = *f*)

Scope

Aria Material

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be

a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output
If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Flux_Output
If specified, the average flux of the given BC/Src is calculated and stored into the global variable name specified as the *flux_output* argument.

Toggle
If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
power_output	"string"	–
flux_output	"string"	–
toggle	"string"	–
f	"string"	–

User_Field

Syntax
Temperature [*{of}* *SpeciesName*] = User_Field ([Power_Output = *power_output*] | [Flux_Output = *flux_output*] | [Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope
Aria Material

Summary
Value from a user field

Description
The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output

If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Flux_Output

If specified, the average flux of the given BC/Src is calculated and stored into the global variable name specified as the *flux_output* argument.

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
power_output	"string"	–
flux_output	"string"	–
toggle	"string"	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Temperature [*{of}* *SpeciesName*] = User_Function ([Power_Output = *power_output*] | [Flux_Output = *flux_output*] | [Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Scope

Aria Material

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output

If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Flux_Output

If specified, the average flux of the given BC/Src is calculated and stored into the global variable name specified as the *flux_output* argument.

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
power_output	"string"	–
flux_output	"string"	–
toggle	"string"	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Temperature [*{of} SpeciesName*] = User_Plugin ([Power_Output = *power_output*] | [Flux_Output = *flux_output*] | [Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output

If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Flux_Output

If specified, the average flux of the given BC/Src is calculated and stored into the global variable name specified as the *flux_output* argument.

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
power_output	“string”	–
flux_output	“string”	–
toggle	“string”	–
Name	“string”	–
plugin_parameters	“string”	–

Tensor Bulk Conductivity

Constant

Syntax

Tensor Bulk Conductivity [*{of}* *SpeciesName*] = Constant ([Xx = *xx*] | [Xy = *xy*] | [Xz = *xz*] | [Yx = *yx*] | [Yy = *yy*] | [Yz = *yz*] | [Zx = *zx*] | [Zy = *zy*] | [Zz = *zz*] | [T11 = *t11*] | [T12 = *t12*] | [T13 = *t13*] | [T21 = *t21*] | [T22 = *t22*] | [T23 = *t23*] | [T31 = *t31*] | [T32 = *t32*] | [T33 = *t33*])

Scope

Aria Material

Summary

Constant tensor value for the bulk conductivity.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
xx	real	0
xy	real	0
xz	real	0
yx	real	0
yy	real	0
yz	real	0
zx	real	0
zy	real	0
zz	real	0
t11	real	0
t12	real	0
t13	real	0
t21	real	0
t22	real	0
t23	real	0
t31	real	0
t32	real	0
t33	real	0

Diagonal

Syntax

Tensor Bulk Conductivity [*{of}* *SpeciesName*] = Diagonal Variable = *variable*

Scope

Aria Material

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	“string”	–

Level_Set_Aligned

Syntax

Tensor Bulk Conductivity [*{of}* *SpeciesName*] = Level_Set_Aligned ([K_Normal = *k_normal*] | [K_Tangent = *k_tangent*] | [Level_Set_Subindex = *level_set_subindex*])

Scope

Aria Material

Summary

Tensor values for the bulk thermal conductivity aligned to a level set field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
k_normal	real	0
k_tangent	real	0
level_set_subindex	integer	0

Tensor_String_Function

Syntax

Tensor Bulk Conductivity [*{of}* *SpeciesName*] = Tensor_String_Function (F_Xx = *f_xx* | F_Yy = *f_yy* | [F_Xy = *f_xy*] | [F_Zz = *f_zz*] | [F_Xz = *f_xz*] | [F_Yz = *f_yz*] | [F_Yx = *f_yx*] | [F_Zx = *f_zx*] | [F_Zy = *f_zy*])

Scope

Aria Material

Summary

Value from a string function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f_xx	“string”	–
f_yy	“string”	–
f_xy	“string”	0.0
f_zz	“string”	–
f_xz	“string”	0.0
f_yz	“string”	0.0
f_yx	“string”	0.0
f_zx	“string”	0.0
f_zy	“string”	0.0

Tensor_String_Function_Symmetric

Syntax

Tensor Bulk Conductivity [*{of}* *SpeciesName*] =
Tensor_String_Function_Symmetric (F_Xx = *f_xx* | F_Yy = *f_yy* | [F_Xy = *f_xy*] | [F_Zz = *f_zz*] | [F_Xz = *f_xz*] | [F_Yz = *f_yz*])

Scope

Aria Material

Summary

Value from a string function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f_xx	“string”	–
f_yy	“string”	–
f_xy	“string”	0.0
f_zz	“string”	–
f_xz	“string”	0.0
f_yz	“string”	0.0

User_Field

Syntax

Tensor Bulk Conductivity [*{of} SpeciesName*] = User_Field (Variable = *variable* | [Restriction = *restriction*])

Scope

Aria Material

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	“string”	–
restriction	“string”	–

Volume_Average

Syntax

Tensor Bulk Conductivity [*{of} SpeciesName*] = Volume_Average ()

Scope

Aria Material

Summary

Bulk conductivity tensor computed from volume average of species bulk conductivities

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–

Tensor Bulk Conductivity Scaling

Constant

Syntax

Tensor Bulk Conductivity Scaling [*{of} SpeciesName*] = Constant *{s | value}* = *s*

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
s	real	–

Copied

Syntax

Tensor Bulk Conductivity Scaling [*{of}* *SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	“string”	–

Exponential

Syntax

Tensor Bulk Conductivity Scaling [*{of}* *SpeciesName*] = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Scope

Aria Material

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Tensor Bulk Conductivity Scaling [*{of}* *SpeciesName*] = Global (Global_Name = *global_name* | [Component = *component*])

Scope

Aria Material

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Tensor Bulk Conductivity Scaling [*{of}* *SpeciesName*] = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Tensor Bulk Conductivity Scaling [*{of}* *SpeciesName*] = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Tensor Bulk Conductivity Scaling [*{of}* *SpeciesName*] =
Scalar_String_Function F = *f*

Scope

Aria Material

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	"string"	–

T_Exponent

Syntax

Tensor Bulk Conductivity Scaling [*{of}* *SpeciesName*] = T_Exponent (Kbulk_Ref = *kbulk_ref* | T_Ref = *t_ref* | N = *n*)

Scope

Aria Material

Summary

Tensor thermal conductivity scaling for bulk material. Nominally this returns $k = k_{ref}\left(\frac{T}{T_{ref}}\right)^n$. If you provide expressions for BULK_DENSITY and SOLID_DENSITY this is also scaled by the ratio of SOLID_DENSITY/BULK_DENSITY

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
kbulk_ref	real	–
t_ref	real	–
n	real	–

User_Field

Syntax

Tensor Bulk Conductivity Scaling [*{of}* *SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
scaling	real	1
global_var	“string”	–
time_function	“string”	–

User_Function

Syntax

Tensor Bulk Conductivity Scaling [*{of}* *SpeciesName*] = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Scope

Aria Material

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1
toggle	"string"	–

User_Plugin

Syntax

Tensor Bulk Conductivity Scaling [*{of}* *SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	“string”	–
plugin_parameters	“string”	–

Tensor Electrical Conductivity

Calore_User_Sub

Syntax

Tensor Electrical Conductivity [*{of}* *SpeciesName*] = Calore_User_Sub (Name = *name* | Type = *type* | [Multiplier = *multiplier*] | [Material_Data_Block = *material_data_block*] | [Data = *data*])

Scope

Aria Material

Summary

Tensor values for temperature dependent electrical conductivity from a calore user subroutine

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
type	“string”	–
multiplier	real	1
material_data_block	“string”	–
data	“string”	–

Constant

Syntax

Tensor Electrical Conductivity [*{of}* *SpeciesName*] = Constant ([Xx = *xx*] | [Xy = *xy*] | [Xz = *xz*] | [Yx = *yx*] | [Yy = *yy*] | [Yz = *yz*] | [Zx = *zx*] | [Zy = *zy*] | [Zz = *zz*] | [T11 = *t11*] | [T12 = *t12*] | [T13 = *t13*] | [T21 = *t21*] | [T22 = *t22*] | [T23 = *t23*] | [T31 = *t31*] | [T32 = *t32*] | [T33 = *t33*])

Scope

Aria Material

Summary

Constant tensor value for the electrical conductivity

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
xx	real	0
xy	real	0
xz	real	0
yx	real	0
yy	real	0
yz	real	0
zx	real	0
zy	real	0
zz	real	0
t11	real	0
t12	real	0
t13	real	0
t21	real	0
t22	real	0
t23	real	0
t31	real	0
t32	real	0
t33	real	0

Tensor_String_Function

Syntax

Tensor Electrical Conductivity [*{of}* *SpeciesName*] = Tensor_String_Function (F_Xx = *f_xx* | F_Yy = *f_yy* | [F_Xy = *f_xy*] | [F_Zz = *f_zz*] | [F_Xz = *f_xz*] | [F_Yz = *f_yz*] | [F_Yx = *f_yx*] | [F_Zx = *f_zx*] | [F_Zy = *f_zy*])

Scope

Aria Material

Summary

Value from a string function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f_xx	“string”	–
f_yy	“string”	–
f_xy	“string”	0.0
f_zz	“string”	–
f_xz	“string”	0.0
f_yz	“string”	0.0
f_yx	“string”	0.0
f_zx	“string”	0.0
f_zy	“string”	0.0

Tensor_String_Function_Symmetric

Syntax

Tensor Electrical Conductivity [*{of}* *SpeciesName*] =
Tensor_String_Function_Symmetric (F_Xx = *f_xx* | F_Yy = *f_yy* | [F_Xy = *f_xy*] | [F_Zz = *f_zz*] | [F_Xz = *f_xz*] | [F_Yz = *f_yz*])

Scope

Aria Material

Summary

Value from a string function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f_xx	“string”	–
f_yy	“string”	–
f_xy	“string”	0.0
f_zz	“string”	–
f_xz	“string”	0.0
f_yz	“string”	0.0

Tensor Electromigration Coefficient

Constant

Syntax

Tensor Electromigration Coefficient [*{of}* *SpeciesName*] = Constant ([Xx = *xx*] | [Xy = *xy*] | [Xz = *xz*] | [Yx = *yx*] | [Yy = *yy*] | [Yz = *yz*] | [Zx = *zx*] | [Zy = *zy*] | [Zz = *zz*] | [T11 = *t11*] | [T12 = *t12*] | [T13 = *t13*] | [T21 = *t21*] | [T22 = *t22*] | [T23 = *t23*] | [T31 = *t31*] | [T32 = *t32*] | [T33 = *t33*])

Scope

Aria Material

Summary

Constant tensor value for the Soret coefficient

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
xx	real	0
xy	real	0
xz	real	0
yx	real	0
yy	real	0
yz	real	0
zx	real	0
zy	real	0
zz	real	0
t11	real	0
t12	real	0
t13	real	0
t21	real	0
t22	real	0
t23	real	0
t31	real	0
t32	real	0
t33	real	0

User_Function

Syntax

Tensor Electromigration Coefficient [*{of}* *SpeciesName*] = User_Function (Name_Xx = *name_xx* | [Name_Yy = *name_yy*] | [Name_Zz = *name_zz*] | X = *x*)

Scope

Aria Material

Summary

Tensor values for the Soret coefficient from a user function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name_xx	"string"	–
name_yy	"string"	–
name_zz	"string"	–
x	"string"	–

Tensor Mass Diffusivity

Constant

Syntax

Tensor Mass Diffusivity [*{of}* *SpeciesName*] = Constant ([Xx = *xx*] | [Xy = *xy*] | [Xz = *xz*] | [Yx = *yx*] | [Yy = *yy*] | [Yz = *yz*] | [Zx = *zx*] | [Zy = *zy*] | [Zz = *zz*] | [T11 = *t11*] | [T12 = *t12*] | [T13 = *t13*] | [T21 = *t21*] | [T22 = *t22*] | [T23 = *t23*] | [T31 = *t31*] | [T32 = *t32*] | [T33 = *t33*])

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
xx	real	0
xy	real	0
xz	real	0
yx	real	0
yy	real	0
yz	real	0
zx	real	0
zy	real	0
zz	real	0
t11	real	0
t12	real	0
t13	real	0
t21	real	0
t22	real	0
t23	real	0
t31	real	0
t32	real	0
t33	real	0

Tensor_String_Function

Syntax

Tensor Mass Diffusivity [*{of}* *SpeciesName*] = Tensor_String_Function (F_Xx = *f_xx* | F_Yy = *f_yy* | [F_Xy = *f_xy*] | [F_Zz = *f_zz*] | [F_Xz = *f_xz*] | [F_Yz = *f_yz*] | [F_Yx = *f_yx*] | [F_Zx = *f_zx*] | [F_Zy = *f_zy*])

Scope

Aria Material

Summary

Value from a string function.

Description

The tensor components are calculated from the user-defined string functions, which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used. Off-diagonal terms are optional and default to 0 if omitted.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f_xx	"string"	–
f_yy	"string"	–
f_xy	"string"	0.0
f_zz	"string"	–
f_xz	"string"	0.0
f_yz	"string"	0.0
f_yx	"string"	0.0
f_zx	"string"	0.0
f_zy	"string"	0.0

Tensor_String_Function_Symmetric

Syntax

Tensor Mass Diffusivity [*{of}* *SpeciesName*] =
Tensor_String_Function_Symmetric (F_Xx = *f_xx* | F_Yy = *f_yy* | [F_Xy = *f_xy*] | [F_Zz = *f_zz*] | [F_Xz = *f_xz*] | [F_Yz = *f_yz*])

Scope

Aria Material

Summary

Value from a string function.

Description

The symmetric tensor components are calculated from the user-defined string functions (only for diagonal and upper triangular terms), which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used. Off-diagonal terms are optional and default to 0 if omitted.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f_xx	"string"	–
f_yy	"string"	–
f_xy	"string"	0.0
f_zz	"string"	–
f_xz	"string"	0.0
f_yz	"string"	0.0

User_Plugin

Syntax

Tensor Mass Diffusivity [*{of}* *SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	“string”	–
plugin_parameters	“string”	–

Tensor Soret Coefficient

Constant

Syntax

Tensor Soret Coefficient [*{of}* *SpeciesName*] = Constant ([Xx = *xx*] | [Xy = *xy*] | [Xz = *xz*] | [Yx = *yx*] | [Yy = *yy*] | [Yz = *yz*] | [Zx = *zx*] | [Zy = *zy*] | [Zz = *zz*] | [T11 = *t11*] | [T12 = *t12*] | [T13 = *t13*] | [T21 = *t21*] | [T22 = *t22*] | [T23 = *t23*] | [T31 = *t31*] | [T32 = *t32*] | [T33 = *t33*])

Scope

Aria Material

Summary

Constant tensor value for the Soret coefficient

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
xx	real	0
xy	real	0
xz	real	0
yx	real	0
yy	real	0
yz	real	0
zx	real	0
zy	real	0
zz	real	0
t11	real	0
t12	real	0
t13	real	0
t21	real	0
t22	real	0
t23	real	0
t31	real	0
t32	real	0
t33	real	0

User_Function

Syntax

Tensor Soret Coefficient [*{of}* *SpeciesName*] = User_Function (Name_Xx = *name_xx* | [Name_Yy = *name_yy*] | [Name_Zz = *name_zz*] | X = *x*)

Scope

Aria Material

Summary

Tensor values for the Soret coefficient from a user function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name_xx	“string”	–
name_yy	“string”	–
name_zz	“string”	–
x	“string”	–

Tensor Species Diffusivity

Constant

Syntax

Tensor Species Diffusivity [*{of}* *SpeciesName*] = Constant ([Xx = *xx*] | [Xy = *xy*] | [Xz = *xz*] | [Yx = *yx*] | [Yy = *yy*] | [Yz = *yz*] | [Zx = *zx*] | [Zy = *zy*] | [Zz = *zz*] | [T11 = *t11*] | [T12 = *t12*] | [T13 = *t13*] | [T21 = *t21*] | [T22 = *t22*] | [T23 = *t23*] | [T31 = *t31*] | [T32 = *t32*] | [T33 = *t33*])

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
xx	real	0
xy	real	0
xz	real	0
yx	real	0
yy	real	0
yz	real	0
zx	real	0
zy	real	0
zz	real	0
t11	real	0
t12	real	0
t13	real	0
t21	real	0
t22	real	0
t23	real	0
t31	real	0
t32	real	0
t33	real	0

Tensor_String_Function

Syntax

Tensor Species Diffusivity [*{of}* *SpeciesName*] = Tensor_String_Function (F_Xx = *f_xx* | F_Yy = *f_yy* | [F_Xy = *f_xy*] | [F_Zz = *f_zz*] | [F_Xz = *f_xz*] | [F_Yz = *f_yz*] | [F_Yx = *f_yx*] | [F_Zx = *f_zx*] | [F_Zy = *f_zy*])

Scope

Aria Material

Summary

Value from a string function.

Description

The tensor components are calculated from the user-defined string functions, which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used. Off-diagonal terms are optional and default to 0 if omitted.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f_xx	“string”	–
f_yy	“string”	–
f_xy	“string”	0.0
f_zz	“string”	–
f_xz	“string”	0.0
f_yz	“string”	0.0
f_yx	“string”	0.0
f_zx	“string”	0.0
f_zy	“string”	0.0

Tensor_String_Function_Symmetric

Syntax

Tensor Species Diffusivity [*{of}* *SpeciesName*] = Tensor_String_Function_Symmetric (F_Xx = *f_xx* | F_Yy = *f_yy* | [F_Xy = *f_xy*] | [F_Zz = *f_zz*] | [F_Xz = *f_xz*] | [F_Yz = *f_yz*])

Scope

Aria Material

Summary

Value from a string function.

Description

The symmetric tensor components are calculated from the user-defined string functions (only for diagonal and upper triangular terms), which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used. Off-diagonal terms are optional and default to 0 if omitted.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f_xx	"string"	–
f_yy	"string"	–
f_xy	"string"	0.0
f_zz	"string"	–
f_xz	"string"	0.0
f_yz	"string"	0.0

User_Plugin

Syntax

Tensor Species Diffusivity [*{of} SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	"string"	–
plugin_parameters	"string"	–

Tensor Species Expansion Coeff

Constant

Syntax

Tensor Species Expansion Coeff [*{of}* *SpeciesName*] = Constant ([Xx = *xx*] | [Xy = *xy*] | [Xz = *xz*] | [Yx = *yx*] | [Yy = *yy*] | [Yz = *yz*] | [Zx = *zx*] | [Zy = *zy*] | [Zz = *zz*] | [T11 = *t11*] | [T12 = *t12*] | [T13 = *t13*] | [T21 = *t21*] | [T22 = *t22*] | [T23 = *t23*] | [T31 = *t31*] | [T32 = *t32*] | [T33 = *t33*])

Scope

Aria Material

Summary

Constant tensor value for the stress coefficient due to species concentration.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
xx	real	0
xy	real	0
xz	real	0
yx	real	0
yy	real	0
yz	real	0
zx	real	0
zy	real	0
zz	real	0
t11	real	0
t12	real	0
t13	real	0
t21	real	0
t22	real	0
t23	real	0
t31	real	0
t32	real	0
t33	real	0

User_Function

Syntax

Tensor Species Expansion Coeff [*{of}* *SpeciesName*] = User_Function (Name_Xx = *name_xx* | [Name_Yy = *name_yy*] | [Name_Zz = *name_zz*] | X = *x*)

Scope

Aria Material

Summary

Tensor values for the species stress coefficient Beta from a user function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name_xx	"string"	–
name_yy	"string"	–
name_zz	"string"	–
x	"string"	–

Tensor Species Mobility

Constant

Syntax

Tensor Species Mobility [*{of}* *SpeciesName*] = Constant ([Xx = *xx*] | [Xy = *xy*] | [Xz = *xz*] | [Yx = *yx*] | [Yy = *yy*] | [Yz = *yz*] | [Zx = *zx*] | [Zy = *zy*] | [Zz = *zz*] | [T11 = *t11*] | [T12 = *t12*] | [T13 = *t13*] | [T21 = *t21*] | [T22 = *t22*] | [T23 = *t23*] | [T31 = *t31*] | [T32 = *t32*] | [T33 = *t33*])

Scope

Aria Material

Summary

Constant tensor value for the electrical conductivity

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
xx	real	0
xy	real	0
xz	real	0
yx	real	0
yy	real	0
yz	real	0
zx	real	0
zy	real	0
zz	real	0
t11	real	0
t12	real	0
t13	real	0
t21	real	0
t22	real	0
t23	real	0
t31	real	0
t32	real	0
t33	real	0

Nernst_Einstein

Syntax

Tensor Species Mobility [*{of}* *SpeciesName*] = Nernst_Einstein (R = *r* | [Temperature_Material_Phase = *temperature_material_phase*])

Scope

Aria Material

Summary

Tensor mobility from diffusivity, M = D/(RT).

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
r	real	–
temperature_material_phase	“string”	–

User_Function

Syntax

Tensor Species Mobility [*{of}* *SpeciesName*] = User_Function (Name_Xx = *name_xx* | [Name_Yy = *name_yy*] | [Name_Zz = *name_zz*] | X = *x*)

Scope

Aria Material

Summary

Tensor values for the species mobility from a user function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name_xx	"string"	–
name_yy	"string"	–
name_zz	"string"	–
x	"string"	–

Tensor Thermal Conductivity

Calore_User_Sub

Syntax

Tensor Thermal Conductivity [*{of}* *SpeciesName*] = Calore_User_Sub (Name = *name* | Type = *type* | [Multiplier = *multiplier*] | [Material_Data_Block = *material_data_block*] | [Data = *data*])

Scope

Aria Material

Summary

Tensor values for the thermal conductivity from a calore user subroutine

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
type	"string"	–
multiplier	real	1
material_data_block	"string"	–
data	"string"	–

Constant

Syntax

Tensor Thermal Conductivity [*{of}* *SpeciesName*] = Constant ([Xx = *xx*] | [Xy = *xy*] | [Xz = *xz*] | [Yx = *yx*] | [Yy = *yy*] | [Yz = *yz*] | [Zx = *zx*] | [Zy = *zy*] | [Zz = *zz*] | [T11 = *t11*] | [T12 = *t12*] | [T13 = *t13*] | [T21 = *t21*] | [T22 = *t22*] | [T23 = *t23*] | [T31 = *t31*] | [T32 = *t32*] | [T33 = *t33*])

Scope

Aria Material

Summary

Constant tensor value for the thermal conductivity

Description

Either

$$k = \begin{bmatrix} \text{XX} & \text{XY} & \text{XZ} \\ \text{YX} & \text{YY} & \text{YZ} \\ \text{ZX} & \text{ZY} & \text{ZZ} \end{bmatrix}$$

or

$$k = \begin{bmatrix} T_{11} & T_{12} & T_{13} \\ T_{21} & T_{22} & T_{23} \\ T_{31} & T_{32} & T_{33} \end{bmatrix}$$

can be used. Any unspecified components of the tensor default to 0.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
xx	real	0
xy	real	0
xz	real	0
yx	real	0
yy	real	0
yz	real	0
zx	real	0
zy	real	0
zz	real	0
t11	real	0
t12	real	0
t13	real	0
t21	real	0
t22	real	0
t23	real	0
t31	real	0
t32	real	0
t33	real	0

Copied

Syntax

Tensor Thermal Conductivity [*{of}* *SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	“string”	–

Level_Set_Aligned

Syntax

Tensor Thermal Conductivity [*{of}* *SpeciesName*] = Level_Set_Aligned ([K_Normal = *k_normal*] | [K_Tangent = *k_tangent*] | [Level_Set_Subindex = *level_set_subindex*])

Scope

Aria Material

Summary

Tensor values for the thermal conductivity aligned to a level set field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
k_normal	real	0
k_tangent	real	0
level_set_subindex	integer	0

Mass_Average

Syntax

Tensor Thermal Conductivity [*{of}* *SpeciesName*] = Mass_Average ()

Scope

Aria Material

Summary

Thermal conductivity computed from mass average of species conductivities

Description

$$k_i j = \frac{\sum_{s \in \text{species}} \left[\text{clip}(Y_s) k_{ij,s}\right]}{\sum_{s \in \text{species}} \text{clip}(Y_s)}$$

where Y_s is the mass fraction of species s, $k_{ij,s}$ the ij component of the conductivity of species s and the clip() function clips values to the bounds [0, 1].

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–

Mesh_Input

Syntax

Tensor Thermal Conductivity [*{of}* *SpeciesName*] = Mesh_Input ([Multiplier = *multiplier*] | [Name = *name*] | [Type = *type*] | [Axis1 = *axis1*] | [Axis2 = *axis2*] | [Tolerance = *tolerance*])

Scope

Aria Material

Summary

Tensor thermal conductivity from the input mesh database.

Description

The "Name" parameter specifies the name of the field on the input mesh to read from. The "Type" parameter must be one of "PRINCIPAL", "SYMMETRIC", or "FULL" depending on the form of the input element field. The "Axis1" and "Axis2" parameters ("Axis1" only for 2D) provide names of vector element fields on the input mesh that define local alignment of principal coordinate system axes for the tensor, they will be used to appropriately rotate the tensor into the XYZ computational frame. The "Multiplier" parameter may be used to uniformly scale all tensor components by a constant factor.

The "Tolerance" parameter is unused, but preserved for backwards compatibility.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
multiplier	real	1
name	"string"	COND
type	"string"	PRINCIPAL
axis1	"string"	FIBER
axis2	"string"	NORMAL
tolerance	real	1e-08

Summed

Syntax

Tensor Thermal Conductivity [*{of}* *SpeciesName*] = Summed Contributions = *contributions*

Scope

Aria Material

Summary

Tensor thermal conductivity as a sum of user-specified conductivities

Description

The “Contributions” parameter should be a quoted list of one or more strings corresponding to Expression names. For example: ‘Contributions=”gas_phase_tensor_thermal_conductivity liquid_phase_tensor_thermal_conductivity”’.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
contributions	“string”	–

Tensor_String_Function

Syntax

Tensor Thermal Conductivity [*{of}* *SpeciesName*] = Tensor_String_Function (F_Xx = *f_xx* | F_Yy = *f_yy* | [F_Xy = *f_xy*] | [F_Zz = *f_zz*] | [F_Xz = *f_xz*] | [F_Yz = *f_yz*] | [F_Yx = *f_yx*] | [F_Zx = *f_zx*] | [F_Zy = *f_zy*])

Scope

Aria Material

Summary

Value from a string function. Off-diagonal terms are optional and default to 0.0 if omitted.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f_xx	“string”	–
f_yy	“string”	–
f_xy	“string”	0.0
f_zz	“string”	–
f_xz	“string”	0.0
f_yz	“string”	0.0
f_yx	“string”	0.0
f_zx	“string”	0.0
f_zy	“string”	0.0

Tensor_String_Function_Symmetric

Syntax

Tensor Thermal Conductivity [*{of}* *SpeciesName*] = Tensor_String_Function_Symmetric (F_Xx = *f_xx* | F_Yy = *f_yy* | [F_Xy = *f_xy*] | [F_Zz = *f_zz*] | [F_Xz = *f_xz*] | [F_Yz = *f_yz*])

Scope

Aria Material

Summary

Value from a string function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f_xx	"string"	–
f_yy	"string"	–
f_xy	"string"	0.0
f_zz	"string"	–
f_xz	"string"	0.0
f_yz	"string"	0.0

User_Function

Syntax

Tensor Thermal Conductivity [*{of}* *SpeciesName*] = User_Function (Name_Xx = *name_xx* | [Name_Yy = *name_yy*] | [Name_Zz = *name_zz*] | X = *x*)

Scope

Aria Material

Summary

Tensor values for the thermal conductivity from a user function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name_xx	"string"	–
name_yy	"string"	–
name_zz	"string"	–
x	"string"	–

Thermal Conductivity

Activation_User_Function

Syntax

Thermal Conductivity [*{of}* *SpeciesName*] = Activation_User_Function ([Background = *background*] | Name = *name* | [Deposit_Name = *deposit_name*])

Scope

Aria Material

Summary

Temperature user function thermal conductivity model with activation field.

Description

$$k = \begin{cases} f_{\text{user}}(T), & \text{activate} = 1 \\ k_{\text{background}} & \text{activate} = 0 \end{cases}$$

where $f_{\text{user}}(T)$ is the specified user function, $k_{\text{background}}$ is set by the "background" parameter and defaults to 0, and activate is the value of the element field, typically set by a directed energy model. Optionally, an additional user function can be specified for the deposited but not activated material $f_{\text{deposit}}(T)$ such that

$$k = \begin{cases} f_{\text{user}}(T), & \text{activate} = 1 \mathit{and} \text{deposit} = 1 \\ f_{\text{deposit}}(T), & \text{activate} = 0 \mathit{and} \text{deposit} = 1 \\ k_{\text{background}} & \text{deposit} = 0 \end{cases}$$

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
background	real	0
name	"string"	–
deposit_name	"string"	–

Calore_User_Sub

Syntax

Thermal Conductivity [*{of} SpeciesName*] = Calore_User_Sub (Name = *name* | Type = *type* | [Multiplier = *multiplier*] | [Material_Data_Block = *material_data_block*] | [Data = *data*] | [Scaling_Field = *scaling_field*])

Scope

Aria Material

Summary

Values from a Calore user subroutine

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
type	"string"	–
multiplier	real	1
material_data_block	"string"	–
data	"string"	–
scaling_field	"string"	–

Cantera

Syntax

Thermal Conductivity [*{of} SpeciesName*] = Cantera [Poffset = *poffset*]

Scope

Aria Material

Summary

Evaluates the thermal conductivity using the Cantera library.

Description

A valid Cantera XML File must be provided elsewhere in the material definition.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
poffset	real	0

Constant

Syntax

Thermal Conductivity [*{of} SpeciesName*] = Constant *{k | value}* = *k*

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
k	real	–

Copied

Syntax

Thermal Conductivity [*{of} SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	“string”	–

Curing_Foam

Syntax

Thermal Conductivity [*{of} SpeciesName*] = Curing_Foam ()

Scope

Aria Material

Summary

A model for the thermal conductivity of curing polymer foams.

Description

$$k = \begin{cases} k_l, & \phi_g <= 0 \\ k_g, & \phi_g >= 1 \\ \frac{\phi_g^{\frac{2}{3}} k_g + k_l (1 - \phi_g^{\frac{2}{3}})}{\frac{k_g (\phi_g^{\frac{2}{3}} - \phi_g)}{k_l} + 1 - \phi_g^{\frac{2}{3}} + \phi_g}, & 0 < \phi_g < 1 \end{cases}$$

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–

Exponential

Syntax

Thermal Conductivity [*{of}* *SpeciesName*] = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Scope

Aria Material

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	“string”	–
constant	real	0
multiplier	real	1
exponent	real	–

Fortran

Syntax

Thermal Conductivity [*{of}* *SpeciesName*] = Fortran ([Multiplier = *multiplier*] | Sub_Name = *sub_name* | [Real_Data = *real_data*] | [Int_Data = *int_data*] | [Resource_Name = *resource_name*] | [Data = *data*])

Scope

Aria Material

Summary

Fortran subroutine thermal conductivity.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
multiplier	real	1
sub_name	"string"	–
real_data	"string"	–
int_data	"string"	–
resource_name	"string"	–
data	"string"	–

From_Prandtl

Syntax

Thermal Conductivity [*{of}* *SpeciesName*] = From_Prandtl ()

Scope

Aria Material

Summary

Thermal conductivity based on the Prandtl number, specific heat, and viscosity.

Description

$$k = \frac{\mu c_p}{\mathrm{Pr}}$$

where μ is the viscosity, c_p the specific heat, and Pr the Prandtl number.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–

Global

Syntax

Thermal Conductivity [*{of} SpeciesName*] = Global (Global_Name = *global_name* | [Component = *component*])

Scope

Aria Material

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
global_name	"string"	–
component	integer	0

Interpolated_Phase_Average

Syntax

Thermal Conductivity [*{of} SpeciesName*] = Interpolated_Phase_Average ([Subindex_A = *subindex_a*] | [Subindex_B = *subindex_b*])

Scope

Aria Material

Summary

Interpolated level set phase averaged thermal conductivity.

Description

See *Phase_Average*, however in this model the evaluation is done based on nodal values and interpolated to the integration points.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
subindex_a	integer	-1
subindex_b	integer	-1

Linear_Temperature_And_Density

Syntax

Thermal Conductivity [*{of}* *SpeciesName*] = Linear_Temperature_And_Density (C_0 = *c_0* | C_Rho = *C_rho* | C_T = *c_t*)

Scope

Aria Material

Summary

Thermal conductivity that is a linear function of both temperature and density.

Description

$$\kappa = C_0 + C_\rho \rho + C_T T$$

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
c_0	real	–
C_rho	real	–
c_t	real	–

Mass_Average

Syntax

Thermal Conductivity [*{of}* *SpeciesName*] = Mass_Average ()

Scope

Aria Material

Summary

Thermal conductivity computed from mass average of species conductivities

Description

$$k = \frac{\sum_{i \in \text{species}} [\text{clip}(Y_i) k_i]}{\sum_{i \in \text{species}} \text{clip}(Y_i)}$$

where Y_i is the mass fraction of species i, k_i the conductivity of species i and the clip() function clips values to the bounds [0, 1].

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–

Nd_Table

Syntax

Thermal Conductivity [*{of} SpeciesName*] = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Optically_Thick

Syntax

Thermal Conductivity [*{of} SpeciesName*] = Optically_Thick (K = *k* | Beta_R = *beta_r* | N = *n*)

Scope

Aria Material

Summary

Thermal conductivity model for optically thick materials.

Description

$$k = K + \frac{16\sigma n^2}{3\beta_r} T^3$$

where K, β_R, and n are the provided parameters, σ the Stefan-Boltzmann constant, and T the temperature.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
k	real	–
beta_r	real	–
n	real	–

Phase_Average

Syntax

Thermal Conductivity [*{of}* *SpeciesName*] = Phase_Average ([Subindex_A = *subindex_a*] | [Subindex_B = *subindex_b*])

Scope

Aria Material

Summary

Level set phase averaged thermal conductivity.

Description

$$k = (1 - H(\phi))k_A + H(\phi)k_B$$

where $H(\phi)$ is the heaviside function evaluated at the level-set variable ϕ, k_A is the thermal conductivity in phase A, and k_B is the thermal conductivity in phase B.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
subindex_a	integer	-1
subindex_b	integer	-1

Polynomial

Syntax

Thermal Conductivity [*{of}* *SpeciesName*] = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Porous_Arithmetic_Mixture

Syntax

Thermal Conductivity [*{of} SpeciesName*] = Porous_Arithmetic_Mixture ([Liquid_Phase_Name = *Liquid_Phase_Name*] | [Gas_Phase_Name = *Gas_Phase_Name*] | [Initial_Saturation = *Initial_Saturation*] | [Melt_Dt = *melt_DT*])

Scope

Aria Material

Summary

Thermal conductivity computed from arithmetic mean of phase properties.

Description

For two-phase porous media problems (solid and liquid phases only):

$$k = (1 - \phi)k_s + \phi k_l$$

where ϕ is the porosity, k_s the solid phase conductivity, and k_l the liquid phase conductivity.

For three-phase porous media problems (solid, liquid, gas):

$$k = (1 - \phi)k_s + \phi S_l k_l + \phi(1 - S_l)k_g$$

where S_l is the liquid saturation and k_g the gas phase conductivity.

Note that the optional "liquid_phase_name" and "gas_phase_name" parameters can be used to specify alternate material phase's supported by Aria for the respective parameters.

The "initial_saturation" and "melt_dt" parameters are for certain thermal battery models, and should be omitted for general use.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Liquid_Phase_Name	"string"	–
Gas_Phase_Name	"string"	–
Initial_Saturation	real	1
melt_DT	real	5

Porous_Bruggeman_Mixture

Syntax

Thermal Conductivity [*{of} SpeciesName*] = Porous_Bruggeman_Mixture ([Liquid_Phase_Name = *Liquid_Phase_Name*] | [Gas_Phase_Name = *Gas_Phase_Name*] | [Initial_Saturation = *Initial_Saturation*] | [Melt_Dt = *melt_DT*] | [Alpha = *alpha*])

Scope

Aria Material

Summary

Thermal conductivity computed with the Bruggeman mixture model

Description

For two-phase porous media problems (solid and liquid phases only):

$$k = (1 - \phi)^\alpha k_s + \phi^\alpha k_l$$

where ϕ is the porosity, k_s the solid phase conductivity, k_l the liquid phase conductivity, and α the optional parameter that defaults to 1.5.

For three-phase porous media problems (solid, liquid, gas):

$$k = (1 - \phi)^\alpha k_s + (\phi S_l)^\alpha k_l + (\phi(1 - S_l))^\alpha k_g$$

where S_l is the liquid saturation and k_g the gas phase conductivity.

Note that the optional "liquid_phase_name" and "gas_phase_name" parameters can be used to specify alternate material phase's supported by Aria for the respective parameters.

The "initial_saturation" and "melt_dt" parameters are for certain thermal battery models, and should be omitted for general use.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Liquid_Phase_Name	"string"	–
Gas_Phase_Name	"string"	–
Initial_Saturation	real	1
melt_DT	real	5
alpha	real	1.5

Porous_Emt_Mixture

Syntax

Thermal Conductivity [*{of} SpeciesName*] = Porous_Emt_Mixture ([Liquid_Phase_Name = *Liquid_Phase_Name*] | [Gas_Phase_Name = *Gas_Phase_Name*] | [Initial_Saturation = *Initial_Saturation*] | [Melt_Dt = *melt_DT*])

Scope

Aria Material

Summary

Thermal conductivity computed effective medium theory model

Description

The effective medium theory model defines a function for averaging the properties of two phases:

$$\mathrm{EMT}(k_1, k_2, f) = \frac{1}{4}\left[x + \sqrt{x^2 + 8k_1k_2}\right]$$
$$x = (3f - 1)k_2 + (3(1 - f) - 1)k_1$$

For two-phase porous media problems (solid and liquid phases only):

$$k = \mathrm{EMT}(k_s, k_l, \phi)$$

where ϕ is the porosity, k_s the solid phase conductivity, and k_l the liquid phase conductivity.

For three-phase porous media problems (solid, liquid, gas):

$$k = \text{EMT}(k_s, \text{EMT}(k_g, k_l, S_l), \phi)$$

where S_l is the liquid saturation and k_g the gas phase conductivity.

Note that the optional "liquid_phase_name" and "gas_phase_name" parameters can be used to specify alternate material phase's supported by Aria for the respective parameters.

The "initial_saturation" and "melt_dt" parameters are for certain thermal battery models, and should be omitted for general use.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Liquid_Phase_Name	"string"	–
Gas_Phase_Name	"string"	–
Initial_Saturation	real	1
melt_DT	real	5

Porous_Geometric_Mixture

Syntax

Thermal Conductivity [*{of} SpeciesName*] = Porous_Geometric_Mixture ([Liquid_Phase_Name = *Liquid_Phase_Name*] | [Gas_Phase_Name = *Gas_Phase_Name*] | [Initial_Saturation = *Initial_Saturation*] | [Melt_Dt = *melt_DT*])

Scope

Aria Material

Summary

Thermal conductivity computed geometric mean of phase properties

Description

For two-phase porous media problems (solid and liquid phases only):

$$k = k_s^{1-\phi} k_l^{\phi}$$

where ϕ is the porosity, k_s the solid phase conductivity, and k_l the liquid phase conductivity.

For three-phase porous media problems (solid, liquid, gas):

$$k = k_s^{1-\phi} k_l^{\phi S_l} k_g^{\phi(1-S_l)}$$

where S_l is the liquid saturation and k_g the gas phase conductivity.

Note that the optional "liquid_phase_name" and "gas_phase_name" parameters can be used to specify alternate material phase's supported by Aria for the respective parameters.

The "initial_saturation" and "melt_dt" parameters are for certain thermal battery models, and should be omitted for general use.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Liquid_Phase_Name	"string"	–
Gas_Phase_Name	"string"	–
Initial_Saturation	real	1
melt_DT	real	5

Porous_Harmonic_Mixture

Syntax

Thermal Conductivity [*{of} SpeciesName*] = Porous_Harmonic_Mixture ([Liquid_Phase_Name = *Liquid_Phase_Name*] | [Gas_Phase_Name = *Gas_Phase_Name*] | [Initial_Saturation = *Initial_Saturation*] | [Melt_Dt = *melt_DT*])

Scope

Aria Material

Summary

Thermal conductivity computed harmonic mean of phase properties

Description

For two-phase porous media problems (solid and liquid phases only):

$$k = \left[\frac{1-\phi}{k_s} + \frac{\phi}{k_l}\right]^{-1}$$

where ϕ is the porosity, k_s the solid phase conductivity, and k_l the liquid phase conductivity.

For three-phase porous media problems (solid, liquid, gas):

$$k = \left[\frac{1-\phi}{k_s} + \frac{\phi S_l}{k_l} + \frac{\phi(1-S_l)}{k_g}\right]^{-1}$$

where S_l is the liquid saturation and k_g the gas phase conductivity.

Note that the optional “liquid_phase_name” and “gas_phase_name” parameters can be used to specify alternate material phase’s supported by Aria for the respective parameters.

The “initial_saturation” and “melt_dt” parameters are for certain thermal battery models, and should be omitted for general use.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Liquid_Phase_Name	“string”	–
Gas_Phase_Name	“string”	–
Initial_Saturation	real	1
melt_DT	real	5

Power_Law

Syntax

Thermal Conductivity [*{of}* *SpeciesName*] = Power_Law (A = *a* | Gamma = *gamma*)

Scope

Aria Material

Summary

Thermal conductivity as a power law in temperature.

Description

$$k = AT^{\gamma}$$

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
a	real	–
gamma	real	–

Saturation_Power_Law

Syntax

Thermal Conductivity [*{of}* *SpeciesName*] = Saturation_Power_Law (K_Dry = *k_dry* | K_Wet = *k_wet* | Power = *power*)

Scope

Aria Material

Summary

Thermal conductivity as a power law in liquid saturation for two phase porous flows.

Description

$$k = \begin{cases} k_{\text{dry}} + S_l^p (k_{\text{wet}} - k_{\text{dry}}), & S_l > 1e-4 \\ k_{\text{dry}}, & s_l <= 1e-4 \end{cases}$$

where S_l is the liquid saturation, and p the provided "power" parameter.

The provided "power" may not be zero, and at least one of "k_dry" and "k_wet" must be non-zero.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
k_dry	real	–
k_wet	real	–
power	real	–

Scalar_String_Function

Syntax

Thermal Conductivity [*{of}* *SpeciesName*] = Scalar_String_Function F = *f*

Scope

Aria Material

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	"string"	–

Summed

Syntax

Thermal Conductivity [*{of}* *SpeciesName*] = Summed Contributions = *contributions*

Scope

Aria Material

Summary

Thermal conductivity as a sum of user-specified contributions.

Description

$$k = \sum_{\text{contributions}} \text{contribution}$$

The "Contributions" parameter should be a quoted list of one or more strings corresponding to Expression names. For example: 'Contributions="gas_phase_thermal_conductivity liquid_phase_thermal_conductivity"'.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
contributions	"string"	–

T_Exponent

Syntax

Thermal Conductivity [*{of}* *SpeciesName*] = T_Exponent (K_Ref = *k_ref* | T_Ref = *t_ref* | N = *n*)

Scope

Aria Material

Summary

Thermal conductivity as a power of normalized temperature

Description

$$k = k_{\text{ref}} * (T/T_{\text{ref}})^n$$

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
k_ref	real	–
t_ref	real	–
n	real	–

Thermal

Syntax

Thermal Conductivity [*{of}* *SpeciesName*] = Thermal ([A = *a*] | [B = *b*] | [C = *c*] | [D = *d*])

Scope

Aria Material

Summary

Thermal conductivity as a cubic polynomial in temperature.

Description

$$k = a + bT + cT^2 + dT^3$$

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
a	real	0
b	real	0
c	real	0
d	real	0

User_Field

Syntax

Thermal Conductivity [*{of}* *SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Thermal Conductivity [*{of} SpeciesName*] = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Scope

Aria Material

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1
toggle	"string"	–

User_Plugin

Syntax

Thermal Conductivity [*{of}* *SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	"string"	–
plugin_parameters	"string"	–

Volume_Average

Syntax

Thermal Conductivity [*{of}* *SpeciesName*] = Volume_Average ()

Scope

Aria Material

Summary

Thermal conductivity computed from volume average of species thermal conductivities.

Description

$$k = \frac{\sum_{i \in \text{species}} [\text{clip}(X_i) k_i]}{\sum_{i \in \text{species}} \text{clip}(X_i)}$$

where X_i is the volume fraction of species i, k_i the conductivity of species i and the clip() function clips values to the bounds [0, 1].

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–

Thermal Diffusivity

Calore_User_Sub

Syntax

Thermal Diffusivity [*{of} SpeciesName*] = Calore_User_Sub (Name = *name* | Type = *type* | [Multiplier = *multiplier*] | [Material_Data_Block = *material_data_block*] | [Data = *data*] | [Scaling_Field = *scaling_field*])

Scope

Aria Material

Summary

Calore user subroutine for the thermal diffusivity

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
type	"string"	–
multiplier	real	1
material_data_block	"string"	–
data	"string"	–
scaling_field	"string"	–

Constant

Syntax

Thermal Diffusivity [*{of} SpeciesName*] = Constant *{d | value}* = *d*

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
d	real	–

Copied

Syntax

Thermal Diffusivity [*{of}* *SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	"string"	–

Exponential

Syntax

Thermal Diffusivity [*{of}* *SpeciesName*] = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Scope

Aria Material

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Thermal Diffusivity [*{of}* *SpeciesName*] = Global (Global_Name = *global_name* | [Component = *component*])

Scope

Aria Material

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
global_name	“string”	–
component	integer	0

Nd_Table

Syntax

Thermal Diffusivity [*{of}* *SpeciesName*] = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Thermal Diffusivity [*{of}* *SpeciesName*] = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Thermal Diffusivity [*{of}* *SpeciesName*] = Scalar_String_Function F = *f*

Scope

Aria Material

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be

a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	"string"	–

User_Field

Syntax

Thermal Diffusivity [*{of}* *SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Thermal Diffusivity [*{of}* *SpeciesName*] = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Scope

Aria Material

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1
toggle	"string"	–

User_Plugin

Syntax

Thermal Diffusivity [*{of} SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	"string"	–
plugin_parameters	"string"	–

Thermal Strain

Constant

Syntax

Thermal Strain [*{of} SpeciesName*] = Constant Value = *value*

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
value	real	–

Copied

Syntax

Thermal Strain [*{of}* *SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	“string”	–

Exponential

Syntax

Thermal Strain [*{of}* *SpeciesName*] = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Scope

Aria Material

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Thermal Strain [*{of}* *SpeciesName*] = Global (Global_Name = *global_name* | [Component = *component*])

Scope

Aria Material

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Thermal Strain [*{of}* *SpeciesName*] = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Thermal Strain [*{of}* *SpeciesName*] = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Thermal Strain [*{of}* *SpeciesName*] = Scalar_String_Function F = *f*

Scope

Aria Material

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	"string"	–

User_Field

Syntax

Thermal Strain [*{of} SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Thermal Strain [*{of} SpeciesName*] = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Scope

Aria Material

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Thermal Strain [*{of}* *SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	“string”	–
plugin_parameters	“string”	–

Thermodynamic Pressure

Constant

Syntax

Thermodynamic Pressure [*{of}* *SpeciesName*] = Constant Value = *value*

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
value	real	–

Copied

Syntax

Thermodynamic Pressure [*{of}* *SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	“string”	–

Exponential

Syntax

Thermodynamic Pressure [*{of}* *SpeciesName*] = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Scope

Aria Material

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Thermodynamic Pressure [*{of} SpeciesName*] = Global (Global_Name = *global_name* | [Component = *component*])

Scope

Aria Material

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Thermodynamic Pressure [*{of} SpeciesName*] = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Thermodynamic Pressure [*{of}* *SpeciesName*] = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Thermodynamic Pressure [*{of}* *SpeciesName*] = Scalar_String_Function F = *f*

Scope

Aria Material

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	"string"	–

User_Field

Syntax

Thermodynamic Pressure [*{of}* *SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Thermodynamic Pressure [*{of}* *SpeciesName*] = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Scope

Aria Material

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Thermodynamic Pressure [*{of}* *SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	“string”	–
plugin_parameters	“string”	–

Tic Drag

Basic

Syntax

Tic Drag [*{of}* *SpeciesName*] = Basic ()

Scope

Aria Material

Summary

Model for drag created in the Theory of Interacting Continua

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–

Tic Drag Coeff

Constant

Syntax

Tic Drag Coeff [*{of} SpeciesName*] = Constant *{alpha_tic | value}* = *alpha_TIC*

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
alpha_TIC	real	–

Copied

Syntax

Tic Drag Coeff [*{of} SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	“string”	–

Exponential

Syntax

Tic Drag Coeff [*{of} SpeciesName*] = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Scope

Aria Material

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Tic Drag Coeff [*{of} SpeciesName*] = Global (Global_Name = *global_name* | [Component = *component*])

Scope

Aria Material

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Tic Drag Coeff [*{of} SpeciesName*] = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Tic Drag Coeff [*{of}* *SpeciesName*] = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Tic Drag Coeff [*{of} SpeciesName*] = Scalar_String_Function F = *f*

Scope

Aria Material

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	"string"	–

User_Field

Syntax

Tic Drag Coeff [*{of}* *SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Tic Drag Coeff [*{of}* *SpeciesName*] = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Scope

Aria Material

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Tic Drag Coeff [*{of}* *SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	“string”	–
plugin_parameters	“string”	–

Time Derivative

Calculated

Syntax

Time Derivative [*{of}* *SpeciesName*] = Calculated (Variable = *variable* | [Max_Order = *max_order*])

Scope

Aria Material

Summary

Calculated time derivative of any expression. This is done by default using a first-order time integration scheme (new - old)/dt To use the same order as the overall solve, set max_order > 1

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
max_order	integer	1

Tortuosity Factor

Bruggeman

Syntax

Tortuosity Factor [*{of}* *SpeciesName*] = Bruggeman ([Exponent = *exponent*] | [Multiplier = *multiplier*] | [Omit_Saturation = *omit_saturation*])

Scope

Aria Material

Summary

Bruggeman model for tortuosity factor. $\tau = \frac{1}{a\phi^n}$ with a default value of $n = 0.5$ and $a = 1$. Set alternate values for them with the 'exponent' and 'multiplier' arguments, respectively.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
exponent	real	0.5
multiplier	real	1
omit_saturation	integer	–

Comiti

Syntax

Tortuosity Factor [*{of}* *SpeciesName*] = Comiti [Min_Porosity = *min_porosity*]

Scope

Aria Material

Summary

This model has been deprecated and removed from the code.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
min_porosity	real	0.001

Constant

Syntax

Tortuosity Factor [*{of}* *SpeciesName*] = Constant Value = *value*

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
value	real	–

Copied

Syntax

Tortuosity Factor [*{of}* *SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	“string”	–

Exponential

Syntax

Tortuosity Factor [*{of}* *SpeciesName*] = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Scope

Aria Material

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Tortuosity Factor [*{of}* *SpeciesName*] = Global (Global_Name = *global_name* | [Component = *component*])

Scope

Aria Material

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
global_name	"string"	–
component	integer	0

Lanfrey

Syntax

Tortuosity Factor [*{of}* *SpeciesName*] = Lanfrey ([Sphericity = *sphericity*] | [Min_Porosity = *min_porosity*])

Scope

Aria Material

Summary

This model has been deprecated and removed from the code.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
sphericity	real	1
min_porosity	real	0.001

Nd_Table

Syntax

Tortuosity Factor [*{of}* *SpeciesName*] = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Tortuosity Factor [*{of}* *SpeciesName*] = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Tortuosity Factor [*{of}* *SpeciesName*] = Scalar_String_Function F = *f*

Scope

Aria Material

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be

a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	"string"	–

User_Field

Syntax

Tortuosity Factor [*{of}* *SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Tortuosity Factor [*{of}* *SpeciesName*] = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Scope

Aria Material

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1
toggle	"string"	–

User_Plugin

Syntax

Tortuosity Factor [*{of} SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	"string"	–
plugin_parameters	"string"	–

Total Internal Energy

Constant

Syntax

Total Internal Energy [*{of} SpeciesName*] = Constant Value = *value*

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
value	real	–

Copied

Syntax

Total Internal Energy [*{of}* *SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	“string”	–

Exponential

Syntax

Total Internal Energy [*{of}* *SpeciesName*] = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Scope

Aria Material

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Total Internal Energy [*{of} SpeciesName*] = Global (Global_Name = *global_name* | [Component = *component*])

Scope

Aria Material

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Total Internal Energy [*{of} SpeciesName*] = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Total Internal Energy [*{of}* *SpeciesName*] = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Porous_Flow

Syntax

Total Internal Energy [*{of}* *SpeciesName*] = Porous_Flow ()

Scope

Aria Material

Summary

Total internal energy for porous flow

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–

Scalar_String_Function

Syntax

Total Internal Energy [*{of}* *SpeciesName*] = Scalar_String_Function F = *f*

Scope

Aria Material

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	"string"	–

User_Field

Syntax

Total Internal Energy [*{of}* *SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Total Internal Energy [*{of}* *SpeciesName*] = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Scope

Aria Material

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1
toggle	"string"	–

User_Plugin

Syntax

Total Internal Energy [*{of} SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	"string"	–
plugin_parameters	"string"	–

Transition Reynolds Number

Correlation

Syntax

Transition Reynolds Number [*{of} SpeciesName*] = Correlation (Rex = *Rex* | [Bulk_Node_Interface = *bulk_node_interface*])

Scope

Aria Material

Summary

Correlation Reynolds number computed from material properties

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Rex	real	–
bulk_node_interface	“string”	–

Transmissivity

Constant

Syntax

Transmissivity [*{of} SpeciesName*] = Constant *{t | value}* = *t*

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
t	real	–

Copied

Syntax

Transmissivity [*{of} SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	"string"	–

Exponential

Syntax

Transmissivity [*{of}* *SpeciesName*] = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Scope

Aria Material

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Transmissivity [*{of}* *SpeciesName*] = Global (Global_Name = *global_name* | [Component = *component*])

Scope

Aria Material

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
global_name	“string”	–
component	integer	0

Nd_Table

Syntax

Transmissivity [*{of}* *SpeciesName*] = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Transmissivity [*{of}* *SpeciesName*] = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Transmissivity [*{of}* *SpeciesName*] = Scalar_String_Function F = *f*

Scope

Aria Material

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	"string"	–

User_Field

Syntax

Transmissivity [*{of}* *SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Transmissivity [*{of}* *SpeciesName*] = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Scope

Aria Material

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1
toggle	"string"	–

User_Plugin

Syntax

Transmissivity [*{of} SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	"string"	–
plugin_parameters	"string"	–

Transport Cross Section

Constant

Syntax

Transport Cross Section [*{of} SpeciesName*] = Constant *{trans | value}* = *trans*

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
trans	real	–

Copied

Syntax

Transport Cross Section [*{of}* *SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	“string”	–

Exponential

Syntax

Transport Cross Section [*{of}* *SpeciesName*] = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Scope

Aria Material

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Transport Cross Section [*{of}* *SpeciesName*] = Global (Global_Name = *global_name* | [Component = *component*])

Scope

Aria Material

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
global_name	"string"	–
component	integer	0

Linearized

Syntax

Transport Cross Section [*{of}* *SpeciesName*] = Linearized (Sigma_0 = *sigma_0* | D_Sigma_D_Rho = *d_sigma_d_rho* | D_Sigma_D_T = *d_sigma_d_t* | T_Sigma_0 = *t_sigma_0* | Rho_Sigma_0 = *rho_sigma_0*)

Scope

Aria Material

Summary

Linearized transport cross section

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
sigma_0	real	–
d_sigma_d_rho	real	–
d_sigma_d_t	real	–
t_sigma_0	real	–
rho_sigma_0	real	–

Nd_Table

Syntax

Transport Cross Section [*{of}* *SpeciesName*] = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Transport Cross Section [*{of}* *SpeciesName*] = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Transport Cross Section [*{of}* *SpeciesName*] = Scalar_String_Function F = *f*

Scope

Aria Material

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be

a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	"string"	–

User_Field

Syntax

Transport Cross Section [*{of}* *SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Transport Cross Section [*{of}* *SpeciesName*] = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Scope

Aria Material

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The

meaning of each option is as follows:

Toggle
If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1
toggle	"string"	–

User_Plugin

Syntax
Transport Cross Section [*{of} SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope
Aria Material

Summary
Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	"string"	–
plugin_parameters	"string"	–

Transported Enthalpy

Material_Phase_Sum

Syntax
Transported Enthalpy [*{of} SpeciesName*] = Material_Phase_Sum ()

Scope
Aria Material

Summary

Provides the mass term for a single temperature porous enthalpy equation by summing enthalpy mass terms across all phases

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–

Porous

Syntax

Transported Enthalpy [*{of}* *SpeciesName*] = Porous ()

Scope

Aria Material

Summary

Provides the mass term for porous enthalpy equation.

Description

For species i and phase p this model provides the porous enthalpy mass term

$$\frac{\partial \phi S_p \rho_{i,p} h_{i,p}}{\partial t},$$

where ϕ is the porosity, S_p is the saturation, $\rho_{i,p}$ is the species density, and $h_{i,p}$ is the species enthalpy in the given phase. If no species is set, this is applied to the general property.

If the optional phase saturation S_p is provided, the pore volume is assumed to be occupied by multiple phases (e.g., liquid / gas) in accordance to their saturations, otherwise it is occupied fully by a single phase. This model is not applicable to the solid phase.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–

Standard

Syntax

Transported Enthalpy [*{of}* *SpeciesName*] = Standard ()

Scope

Aria Material

Summary

Provides the mass term for the porous enthalpy equation.

Description

For species i and phase p this model provides the porous enthalpy mass term

$$\frac{\partial \rho_{i,p} h_{i,p}}{\partial t},$$

where $\rho_{i,p}$ is the species density and $h_{i,p}$ is the species enthalpy for the given phase.

This model can be used to provide the mass term for the solid phase porous enthalpy equation, where note the density provided must be solid mass per total volume.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–

Trapped Solvent Number Density

Exponential

Syntax

Trapped Solvent Number Density [*{of}* *SpeciesName*] = Exponential (A = *a* | B = *b* | C = *c* | Avogadro = *avogadro*)

Scope

Aria Material

Summary

Strain-dependent trapped solvent number density

Description

Calculates trapped solvent number density as a function of plastic strain (ϵ_p) as:

$$N = \frac{1}{N_A} 10^{A - B exp(-C\epsilon_p)}$$

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
a	real	–
b	real	–
c	real	–
avogadro	real	–

Turbulence Dissipation Rate

Constant

Syntax

Turbulence Dissipation Rate [*{of}* *SpeciesName*] = Constant Value = *value*

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
value	real	–

Copied

Syntax

Turbulence Dissipation Rate [*{of}* *SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	“string”	–

Exponential

Syntax

Turbulence Dissipation Rate [*{of}* *SpeciesName*] = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Scope

Aria Material

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Turbulence Dissipation Rate [*{of}* *SpeciesName*] = Global (Global_Name = *global_name* | [Component = *component*])

Scope

Aria Material

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Turbulence Dissipation Rate [*{of} SpeciesName*] = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Turbulence Dissipation Rate [*{of} SpeciesName*] = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Turbulence Dissipation Rate [*{of} SpeciesName*] = Scalar_String_Function F = *f*

Scope

Aria Material

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	"string"	–

User_Field

Syntax

Turbulence Dissipation Rate [*{of} SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Turbulence Dissipation Rate [*{of} SpeciesName*] = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Scope

Aria Material

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Turbulence Dissipation Rate [*{of}* *SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	“string”	–
plugin_parameters	“string”	–

Turbulence Dissipation Rate Diffusive Flux

Basic

Syntax

Turbulence Dissipation Rate Diffusive Flux [*{of}* *SpeciesName*] = Basic ()

Scope

Aria Material

Summary

Basic model of turbulence dissipation rate diffusive flux

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–

Turbulent Bulk Viscosity

Constant

Syntax

Turbulent Bulk Viscosity [*{of} SpeciesName*] = Constant *{kappa | value}* = *kappa*

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
kappa	real	–

Copied

Syntax

Turbulent Bulk Viscosity [*{of} SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	“string”	–

Exponential

Syntax

Turbulent Bulk Viscosity [*{of} SpeciesName*] = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Scope

Aria Material

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Turbulent Bulk Viscosity [*{of} SpeciesName*] = Global (Global_Name = *global_name* | [Component = *component*])

Scope

Aria Material

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Turbulent Bulk Viscosity [*{of} SpeciesName*] = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Turbulent Bulk Viscosity [*{of}* *SpeciesName*] = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Turbulent Bulk Viscosity [*{of}* *SpeciesName*] = Scalar_String_Function F = *f*

Scope

Aria Material

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	"string"	–

User_Field

Syntax

Turbulent Bulk Viscosity [*{of}* *SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Turbulent Bulk Viscosity [*{of}* *SpeciesName*] = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Scope

Aria Material

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Turbulent Bulk Viscosity [*{of}* *SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	“string”	–
plugin_parameters	“string”	–

Turbulent Correlation Heat Transfer Coefficient

Correlation

Syntax

Turbulent Correlation Heat Transfer Coefficient [*{of}* *SpeciesName*] = Correlation (Number = *number* | [Rex = *Rex*] | [Bulk_Node_Interface = *bulk_node_interface*])

Scope

Aria Material

Summary

Correlation based heat transfer coefficient

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
number	integer	–
Rex	real	0
bulk_node_interface	“string”	–

Turbulent Energy Diffusive Flux

Gradient_Transport

Syntax

Turbulent Energy Diffusive Flux [*{of}* *SpeciesName*] = Gradient_Transport ()

Scope

Aria Material

Summary

Gradient transport model of turbulent energy diffusive flux

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–

Turbulent Kinetic Energy

Constant

Syntax

Turbulent Kinetic Energy [*{of}* *SpeciesName*] = Constant Value = *value*

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
value	real	–

Copied

Syntax

Turbulent Kinetic Energy [*{of}* *SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	"string"	–

Exponential

Syntax

Turbulent Kinetic Energy [*{of}* *SpeciesName*] = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Scope

Aria Material

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Turbulent Kinetic Energy [*{of}* *SpeciesName*] = Global (Global_Name = *global_name* | [Component = *component*])

Scope

Aria Material

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
global_name	“string”	–
component	integer	0

Nd_Table

Syntax

Turbulent Kinetic Energy [*{of}* *SpeciesName*] = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Turbulent Kinetic Energy [*{of}* *SpeciesName*] = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Turbulent Kinetic Energy [*{of}* *SpeciesName*] = Scalar_String_Function F = *f*

Scope

Aria Material

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be

a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	"string"	–

User_Field

Syntax

Turbulent Kinetic Energy [*{of}* *SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Turbulent Kinetic Energy [*{of}* *SpeciesName*] = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Scope

Aria Material

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1
toggle	"string"	–

User_Plugin

Syntax

Turbulent Kinetic Energy [*{of} SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	"string"	–
plugin_parameters	"string"	–

Wall_Function

Syntax

Turbulent Kinetic Energy [*{of} SpeciesName*] = Wall_Function Wall_Friction_Factor = *wall_friction_factor*

Scope

Aria Material

Summary

Turbulent kinetic energy using wall function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
wall_friction_factor	real	–

Turbulent Kinetic Energy Density

Constant

Syntax

Turbulent Kinetic Energy Density [*{of}* *SpeciesName*] = Constant Value = *value*

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
value	real	–

Copied

Syntax

Turbulent Kinetic Energy Density [*{of}* *SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	“string”	–

Exponential

Syntax

Turbulent Kinetic Energy Density [*{of}* *SpeciesName*] = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Scope

Aria Material

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Turbulent Kinetic Energy Density [*{of}* *SpeciesName*] = Global (Global_Name = *global_name* | [Component = *component*])

Scope

Aria Material

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Turbulent Kinetic Energy Density [*{of}* *SpeciesName*] = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Turbulent Kinetic Energy Density [*{of}* *SpeciesName*] = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Turbulent Kinetic Energy Density [*{of}* *SpeciesName*] = Scalar_String_Function F = *f*

Scope

Aria Material

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	"string"	–

User_Field

Syntax

Turbulent Kinetic Energy Density [*{of}* *SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Turbulent Kinetic Energy Density [*{of}* *SpeciesName*] = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Scope

Aria Material

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Turbulent Kinetic Energy Density [*{of}* *SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	“string”	–
plugin_parameters	“string”	–

Turbulent Kinetic Energy Diffusive Flux

Basic

Syntax

Turbulent Kinetic Energy Diffusive Flux [*{of}* *SpeciesName*] = Basic ()

Scope

Aria Material

Summary

Basic model of turbulent kinetic energy diffusive flux

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–

Turbulent Mass Diffusivity

Constant

Syntax

Turbulent Mass Diffusivity [*{of} SpeciesName*] = Constant *{dt | value}* = *Dt*

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Dt	real	–

Copied

Syntax

Turbulent Mass Diffusivity [*{of} SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	“string”	–

Exponential

Syntax

Turbulent Mass Diffusivity [*{of} SpeciesName*] = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Scope

Aria Material

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

From_Schmidt

Syntax

Turbulent Mass Diffusivity [*{of}* *SpeciesName*] = From_Schmidt ()

Scope

Aria Material

Summary

Turbulent mass diffusivity from Schmidt number and viscosity

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–

Global

Syntax

Turbulent Mass Diffusivity [*{of}* *SpeciesName*] = Global (Global_Name = *global_name* | [Component = *component*])

Scope

Aria Material

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
global_name	“string”	–
component	integer	0

Nd_Table

Syntax

Turbulent Mass Diffusivity [*{of} SpeciesName*] = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Turbulent Mass Diffusivity [*{of} SpeciesName*] = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Turbulent Mass Diffusivity [*{of} SpeciesName*] = Scalar_String_Function F = *f*

Scope

Aria Material

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	"string"	–

User_Field

Syntax

Turbulent Mass Diffusivity [*{of}* *SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Turbulent Mass Diffusivity [*{of}* *SpeciesName*] = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Scope

Aria Material

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1
toggle	"string"	–

User_Plugin

Syntax

Turbulent Mass Diffusivity [*{of} SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	"string"	–
plugin_parameters	"string"	–

Turbulent Mass Fraction Diffusive Flux

Gradient_Transport

Syntax

Turbulent Mass Fraction Diffusive Flux [*{of} SpeciesName*] = Gradient_Transport ()

Scope

Aria Material

Summary

Gradient transport model of turbulent mass fraction diffusive flux

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–

Turbulent Mixture Fraction Diffusive Flux

Gradient_Transport

Syntax

Turbulent Mixture Fraction Diffusive Flux [*{of}* *SpeciesName*] = Gradient_Transport ()

Scope

Aria Material

Summary

Gradient transport model of turbulent mixture fraction diffusive flux

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–

Turbulent Mixture Fraction Diffusivity

Constant

Syntax

Turbulent Mixture Fraction Diffusivity [*{of}* *SpeciesName*] = Constant *{dt | value}* = *Dt*

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Dt	real	–

Copied

Syntax

Turbulent Mixture Fraction Diffusivity [*{of}* *SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	"string"	–

Exponential

Syntax

Turbulent Mixture Fraction Diffusivity [*{of}* *SpeciesName*] = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Scope

Aria Material

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

From_Schmidt

Syntax

Turbulent Mixture Fraction Diffusivity [*{of}* *SpeciesName*] = From_Schmidt ()

Scope

Aria Material

Summary

Turbulent mixture fraction diffusivity from trb Schmidt number and trb viscosity

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–

Global

Syntax

Turbulent Mixture Fraction Diffusivity [*{of}* *SpeciesName*] = Global (Global_Name = *global_name* | [Component = *component*])

Scope

Aria Material

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
global_name	“string”	–
component	integer	0

Nd_Table

Syntax

Turbulent Mixture Fraction Diffusivity [*{of}* *SpeciesName*] = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Turbulent Mixture Fraction Diffusivity [*{of}* *SpeciesName*] = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Turbulent Mixture Fraction Diffusivity [*{of}* *SpeciesName*] = Scalar_String_Function F = *f*

Scope

Aria Material

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	"string"	–

User_Field

Syntax

Turbulent Mixture Fraction Diffusivity [*{of}* *SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Turbulent Mixture Fraction Diffusivity [*{of}* *SpeciesName*] = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Scope

Aria Material

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Turbulent Mixture Fraction Diffusivity [*{of}* *SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	“string”	–
plugin_parameters	“string”	–

Turbulent Momentum Stress

Constant

Syntax

Turbulent Momentum Stress [*{of}* *SpeciesName*] = Constant ([Xx = *xx*] | [Xy = *xy*] | [Xz = *xz*] | [Yx = *yx*] | [Yy = *yy*] | [Yz = *yz*] | [Zx = *zx*] | [Zy = *zy*] | [Zz = *zz*] | [T11 = *t11*] | [T12 = *t12*] | [T13 = *t13*] | [T21 = *t21*] | [T22 = *t22*] | [T23 = *t23*] | [T31 = *t31*] | [T32 = *t32*] | [T33 = *t33*])

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
xx	real	0
xy	real	0
xz	real	0
yx	real	0
yy	real	0
yz	real	0
zx	real	0
zy	real	0
zz	real	0
t11	real	0
t12	real	0
t13	real	0
t21	real	0
t22	real	0
t23	real	0
t31	real	0
t32	real	0
t33	real	0

Formal_Newtonian_Isotropic

Syntax

Turbulent Momentum Stress [*{of}* *SpeciesName*] =
Formal_Newtonian_Isotropic [Reference_Frame = *reference_frame*]

Scope

Aria Material

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
reference_frame	“string”	MOVING

Incompressible_Newtonian_Isotropic

Syntax

Turbulent Momentum Stress [*{of}* *SpeciesName*] =
Incompressible_Newtonian_Isotropic [Reference_Frame = *reference_frame*]

Scope

Aria Material

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
reference_frame	"string"	MOVING

Newtonian_Dilational_Isotropic

Syntax

Turbulent Momentum Stress [*{of}* *SpeciesName*] =
Newtonian_Dilational_Isotropic [Reference_Frame = *reference_frame*]

Scope

Aria Material

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
reference_frame	"string"	MOVING

Newtonian_Isotropic

Syntax

Turbulent Momentum Stress [*{of}* *SpeciesName*] = Newtonian_Isotropic
[Reference_Frame = *reference_frame*]

Scope

Aria Material

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
reference_frame	"string"	MOVING

Tensor_String_Function

Syntax

Turbulent Momentum Stress [*{of}* *SpeciesName*] = Tensor_String_Function (F_Xx = *f_xx* | F_Yy = *f_yy* | [F_Xy = *f_xy*] | [F_Zz = *f_zz*] | [F_Xz = *f_xz*] | [F_Yz = *f_yz*] | [F_Yx = *f_yx*] | [F_Zx = *f_zx*] | [F_Zy = *f_zy*])

Scope

Aria Material

Summary

Value from a string function.

Description

The tensor components are calculated from the user-defined string functions, which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used. Off-diagonal terms are optional and default to 0 if omitted.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f_xx	“string”	–
f_yy	“string”	–
f_xy	“string”	0.0
f_zz	“string”	–
f_xz	“string”	0.0
f_yz	“string”	0.0
f_yx	“string”	0.0
f_zx	“string”	0.0
f_zy	“string”	0.0

Tensor_String_Function_Symmetric

Syntax

Turbulent Momentum Stress [*{of}* *SpeciesName*] = Tensor_String_Function_Symmetric (F_Xx = *f_xx* | F_Yy = *f_yy* | [F_Xy = *f_xy*] | [F_Zz = *f_zz*] | [F_Xz = *f_xz*] | [F_Yz = *f_yz*])

Scope

Aria Material

Summary

Value from a string function.

Description

The symmetric tensor components are calculated from the user-defined string functions (only for diagonal and upper triangular terms), which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used. Off-diagonal terms are optional and default to 0 if omitted.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f_xx	"string"	–
f_yy	"string"	–
f_xy	"string"	0.0
f_zz	"string"	–
f_xz	"string"	0.0
f_yz	"string"	0.0

User_Plugin

Syntax

Turbulent Momentum Stress [*{of} SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	"string"	–
plugin_parameters	"string"	–

Turbulent Prandtl Number

Constant

Syntax

Turbulent Prandtl Number [*{of} SpeciesName*] = Constant *{pr_t | value}* = *Pr_t*

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Pr_t	real	–

Copied

Syntax

Turbulent Prandtl Number [*{of} SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	“string”	–

Exponential

Syntax

Turbulent Prandtl Number [*{of} SpeciesName*] = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Scope

Aria Material

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Turbulent Prandtl Number [*{of} SpeciesName*] = Global (Global_Name = *global_name* | [Component = *component*])

Scope

Aria Material

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Turbulent Prandtl Number [*{of} SpeciesName*] = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Turbulent Prandtl Number [*{of}* *SpeciesName*] = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Turbulent Prandtl Number [*{of}* *SpeciesName*] = Scalar_String_Function F = *f*

Scope

Aria Material

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	"string"	–

User_Field

Syntax

Turbulent Prandtl Number [*{of}* *SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Turbulent Prandtl Number [*{of}* *SpeciesName*] = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Scope

Aria Material

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Turbulent Prandtl Number [*{of} SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	“string”	–
plugin_parameters	“string”	–

Turbulent Schmidt Number

Constant

Syntax

Turbulent Schmidt Number [*{of} SpeciesName*] = Constant *{sc_t | value}* = *Sc_t*

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Sc_t	real	–

Copied

Syntax

Turbulent Schmidt Number [*{of}* *SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	“string”	–

Exponential

Syntax

Turbulent Schmidt Number [*{of}* *SpeciesName*] = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Scope

Aria Material

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Turbulent Schmidt Number [*{of} SpeciesName*] = Global (Global_Name = *global_name* | [Component = *component*])

Scope

Aria Material

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Turbulent Schmidt Number [*{of} SpeciesName*] = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Turbulent Schmidt Number [*{of}* *SpeciesName*] = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Turbulent Schmidt Number [*{of} SpeciesName*] = Scalar_String_Function F = *f*

Scope

Aria Material

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	"string"	–

User_Field

Syntax

Turbulent Schmidt Number [*{of}* *SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Turbulent Schmidt Number [*{of}* *SpeciesName*] = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Scope

Aria Material

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Turbulent Schmidt Number [*{of}* *SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	“string”	–
plugin_parameters	“string”	–

Turbulent String Function Heat Transfer Coefficient

Correlation

Syntax

Turbulent String Function Heat Transfer Coefficient [*{of}* *SpeciesName*] = Correlation (String_Function = *string_function* | [Bulk_Node_Interface = *bulk_node_interface*])

Scope

Aria Material

Summary

String function that can be used in place of a numbered heat transfer correlation coefficient.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
string_function	"string"	–
bulk_node_interface	"string"	FALSE

Turbulent Thermal Conductivity

From_Prandtl

Syntax

Turbulent Thermal Conductivity [*{of}* *SpeciesName*] = From_Prandtl ()

Scope

Aria Material

Summary

Turbulent thermal conductivity when closing energy term by grad T

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–

Turbulent Thermal Diffusivity

From_Prandtl

Syntax

Turbulent Thermal Diffusivity [*{of}* *SpeciesName*] = From_Prandtl ()

Scope

Aria Material

Summary

?

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–

U

User Expression

Binary_State

Syntax

User Expression ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) = Binary_State (User_Tag = *user_tag* | Location = *location* | [Initial_State = *initial_state*] | [Activation_Function = *activation_function*] | [Deactivation_Function = *deactivation_function*] | [Equations = *Equations. . .*])

Scope

Aria Material

Summary

Binary State user expression

Description

Provides an expression that represents the active/inactive state with criteria for transition. Parameters are:

User_Tag - Name of expression and field to provide

Location - Either ELEMENT to create an element field/expression or NODE to create a nodal field/expression

Initial State - Either ON/ACTIVE or OFF/INACTIVE

Activation Function - String function criteria that will set field state to active

Deactivation Function - String function criteria that will set field state to inactive

Either an activation function, deactivation function, or both must be provided.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
user_tag	“string”	–
location	“string”	–
initial_state	“string”	–
activation_function	“string”	–
deactivation_function	“string”	–
Equations	string. . .	–

Constant

Syntax

User Expression ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) = Constant (User_Tag = *user_tag* | Value = *value* | [Equations = *Equations. . .*])

Scope

Aria Material

Summary

Constant user expression

Description

Provides an expression with the name given by the user_tag parameter using the provided constant value. The provided user tag should not include operator, subindex, phase, component, or material phase prefixes or suffixes. See *General Naming Convention* for more detail about those.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
user_tag	"string"	–
value	real	–
Equations	string. . .	–

Copied

Syntax

User Expression ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) = Copied (User_Tag = *user_tag* | Source = *source* | [Model = *model*] | [Source_Species = *source_species*] | [Source_Phase_Label = *source_phase_label*] | [Source_Material_Phase_Label = *source_material_phase_label*] | [Equations = *Equations. . .*])

Scope

Aria Material

Summary

Copied user expression

Description

Provides an expression that is a copy of an existing expression. If none of the

optional “SOURCE_SPECIES”, “SOURCE_PHASE_LABEL” and “SOURCE_MATERIAL_PHASE_LABEL” parameters are provided then the provided “SOURCE” parameter will be decoded to extract the species name, phase, and material phase. For example “SOLID_PHASE_POROSITY” would be decoded to correspond to the POROSITY in the solid phase. However, if any of the optional “SOURCE_*” parameters are provided then the “SOURCE” parameter is treated only as the expression name.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
user_tag	“string”	–
source	“string”	–
model	“string”	–
source_species	“string”	–
source_phase_label	“string”	–
source_material_phase_label	“string”	–
Equations	string. . .	–

Cross_Product

Syntax

User Expression ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) = Cross_Product (User_Tag = *user_tag* | U = *u* | V = *v* | [Equations = *Equations. . .*])

Scope

Aria Material

Summary

Cross product user expression

Description

Takes the cross product of two vectors (u and v) and produces a vector, X where $X = u \times v$

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
user_tag	"string"	–
u	"string"	–
v	"string"	–
Equations	string. . .	–

Dot_Product

Syntax

User Expression ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) = Dot_Product (User_Tag = *user_tag* | A = *a* | B = *b* | [Equations = *Equations. . .*])

Scope

Aria Material

Summary

Dot product user expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
user_tag	"string"	–
a	"string"	–
b	"string"	–
Equations	string. . .	–

Exponential

Syntax

User Expression ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) = Exponential (User_Tag = *user_tag* | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent* | [Equations = *Equations. . .*])

Scope

Aria Material

Summary

Exponential user expression

Description

Provides an expression with the name given by the user_tag parameter using the provided exponential function. The provided user tag should not include operator, subindex, phase, component, or material phase prefixes or suffixes. See *General Naming Convention* for more detail about those. The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
user_tag	“string”	–
variable	“string”	–
constant	real	0
multiplier	real	1
exponent	real	–
Equations	string. . .	–

From_Scalar_Elem_Field

Syntax

User Expression ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) = From_Scalar_Elem_Field (User_Tag = *user_tag* | Name = *name* | [Offset = *offset*] | [Equations = *Equations. . .*])

Scope

Aria Material

Summary

User expression from a scalar element field

Description

Provides an expression from element rank, edge rank, or face rank scalar fields. Use the offset to index into one of (potentially) several scalars stored at a mesh entity.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
user_tag	"string"	–
name	"string"	–
offset	integer	0
Equations	string. . .	–

Global

Syntax

User Expression ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) = Global (User_Tag = *user_tag* | Global_Name = *global_name* | [Component = *component*] | [Equations = *Equations. . .*])

Scope

Aria Material

Summary

Global user expression

Description

Provides an expression with the name given by the user_tag parameter using the provided global variable. The provided user tag should not include operator, subindex, phase, component, or material phase prefixes or suffixes. See *General Naming Convention* for more detail about those.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
user_tag	"string"	–
global_name	"string"	–
component	integer	0
Equations	string. . .	–

Nd_Table

Syntax

User Expression ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) = Nd_Table (User_Tag = *user_tag* | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*] | [Equations = *Equations. . .*])

Scope

Aria Material

Summary

User expression from a multi-dimensional table.

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
user_tag	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–
Equations	string. . .	–

Outer_Product

Syntax

User Expression ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) = Outer_Product (User_Tag = *user_tag* | U = *u* | V = *v* | [Equations = *Equations. . .*])

Scope

Aria Material

Summary

Cross product user expression

Description

Takes the outer product of two vectors (u and v) and produces a tensor, by multiplying each element of u by each element of v

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
user_tag	"string"	–
u	"string"	–
v	"string"	–
Equations	string. . .	–

Polynomial

Syntax

User Expression ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) = Polynomial (User_Tag = *user_tag* | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*] | [Equations = *Equations. . .*])

Scope

Aria Material

Summary

Polynomial user expression

Description

Provides an expression with the name given by the user_tag parameter using the provided polynomial function. The provided user tag should not include operator, subindex, phase, component, or material phase prefixes or suffixes. See *General Naming Convention* for more detail about those. The polynomial value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
user_tag	"string"	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0
Equations	string. . .	–

Rotation_Tensor

Syntax

User Expression ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) = Rotation_Tensor (User_Tag = *user_tag* | Theta = *theta* | [Nx = *nx*] | [Ny = *ny*] | [Nz = *nz*] | [Equations = *Equations. . .*])

Scope

Aria Material

Summary

Rotation tensor user expression

Description

Calculates a rotation tensor around a defined axis [nx, ny, nz] by an angle expression 'theta'. In 2D the axis is always [0, 0, 1]

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
user_tag	"string"	–
theta	"string"	–
nx	real	0
ny	real	0
nz	real	0
Equations	string. . .	–

Scalar_String_Function

Syntax

User Expression ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) = Scalar_String_Function (User_Tag = *user_tag* | F = *f* | [Equations = *Equations. . .*])

Scope

Aria Material

Summary

String function user expression

Description

Provides an expression with the name given by the user_tag parameter using the provided string function. The provided user tag should not include operator, subindex, phase, component, or material phase prefixes or suffixes. See *General Naming Convention* for more detail about those.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
user_tag	"string"	–
f	"string"	–
Equations	string. . .	–

Scalar_Tensor_Product

Syntax

User Expression ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) = Scalar_Tensor_Product (User_Tag = *user_tag* | Tensor = *tensor* | [Scalars = *scalars*] | [Multiplier = *multiplier*] | [Equations = *Equations. . .*])

Scope

Aria Material

Summary

Scalar tensor product

Description

Multiplies a tensor by a list of provided scalars (variables) and a multiplier (constant)

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
user_tag	“string”	–
tensor	“string”	–
scalars	“string”	–
multiplier	real	1
Equations	string. . .	–

Scalar_Vector_Product

Syntax

User Expression ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) = Scalar_Vector_Product (User_Tag = *user_tag* | Vector = *vector* | [Scalars = *scalars*] | [Multiplier = *multiplier*] | [Equations = *Equations. . .*])

Scope

Aria Material

Summary

Scalar vector product

Description

Multiplies a vector by a list of provided scalars (variables) and a multiplier (constant)

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
user_tag	"string"	–
vector	"string"	–
scalars	"string"	–
multiplier	real	1
Equations	string. . .	–

Tensor_Determinant

Syntax

User Expression ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) = Tensor_Determinant (User_Tag = *user_tag* | Tensor = *tensor* | [Equations = *Equations. . .*])

Scope

Aria Material

Summary

Tensor determinant

Description

Computes the determinant of the provided tensor

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
user_tag	"string"	–
tensor	"string"	–
Equations	string. . .	–

Tensor_String_Function

Syntax

User Expression ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) = Tensor_String_Function (User_Tag = *user_tag* | F_Xx = *f_xx* | F_Yy = *f_yy* | [F_Xy = *f_xy*] | [F_Zz = *f_zz*] | [F_Xz = *f_xz*] | [F_Yz = *f_yz*] | [F_Yx = *f_yx*] | [F_Zx = *f_zx*] | [F_Zy = *f_zy*] | [Equations = *Equations. . .*])

Scope

Aria Material

Summary

Tensor string function user expression

Description

Provides an expression with the name given by the user_tag parameter using the provided string function. The provided user tag should not include operator, subindex, phase, component, or material phase prefixes or suffixes. See *General Naming Convention* for more detail about those.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
user_tag	“string”	–
f_xx	“string”	–
f_yy	“string”	–
f_xy	“string”	0.0
f_zz	“string”	–
f_xz	“string”	0.0
f_yz	“string”	0.0
f_yx	“string”	0.0
f_zx	“string”	0.0
f_zy	“string”	0.0
Equations	string. . .	–

Tensor_String_Function_Symmetric

Syntax

User Expression ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) = Tensor_String_Function_Symmetric (User_Tag = *user_tag* | F_Xx = *f_xx* | F_Yy = *f_yy* | [F_Xy = *f_xy*] | [F_Zz = *f_zz*] | [F_Xz = *f_xz*] | [F_Yz = *f_yz*] | [Equations = *Equations. . .*])

Scope

Aria Material

Summary

Symmetric tensor string function user expression

Description

Provides an expression with the name given by the user_tag parameter using the provided string function. The provided user tag should not include operator, subindex, phase, component, or material phase prefixes or suffixes. See *General Naming Convention* for more detail about those.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
user_tag	“string”	–
f_xx	“string”	–
f_yy	“string”	–
f_xy	“string”	0.0
f_zz	“string”	–
f_xz	“string”	0.0
f_yz	“string”	0.0
Equations	string. . .	–

Tensor_Sum

Syntax

User Expression ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) = Tensor_Sum (User_Tag = *user_tag* | Contributions = *contributions* | [Equations = *Equations. . .*])

Scope

Aria Material

Summary

Tensor sum

Description

Summation of a list of tensor and/or scalar contributions. Scalar sums are applied to all tensor components

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
user_tag	"string"	–
contributions	"string"	–
Equations	string. . .	–

Tensor_Tensor_Product

Syntax

User Expression ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) = Tensor_Tensor_Product (User_Tag = *user_tag* | A = *a* | B = *b* | [Equations = *Equations. . .*])

Scope

Aria Material

Summary

Calculates the product of two tensors

Description

Calculates the resulting tensor, C, from the tensor multiplication of A and B ($AB = C$)

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
user_tag	"string"	–
a	"string"	–
b	"string"	–
Equations	string. . .	–

Tensor_Vector_Product

Syntax

User Expression ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) = Tensor_Vector_Product (User_Tag = *user_tag* | Vector = *vector* | Tensor = *tensor* | [Scale = *scale*] | [Equations = *Equations. . .*])

Scope

Aria Material

Summary

Tensor vector product

Description

Calculates a vector resulting from a tensor-vector product with an optional scaling factor.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
user_tag	"string"	–
vector	"string"	–
tensor	"string"	–
scale	real	1
Equations	string. . .	–

Transform_Tensor

Syntax

User Expression ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) = Transform_Tensor (User_Tag = *user_tag* | U = *u* | R = *r* | [Equations = *Equations. . .*])

Scope

Aria Material

Summary

Cross product user expression

Description

Does a tensor transformation $U^{'} = R^{T}UR$ where R is an orthogonal rotation matrix and U is a tensor. If multiple rotation matrices are provided e.g. R ="Rz Ry Rx", then the rotations will be applied in order, e.g. $R = R_x R_y R_z$.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
user_tag	"string"	–
u	"string"	–
r	"string"	–
Equations	string. . .	–

User_Field

Syntax

User Expression ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) = User_Field (User_Tag = *user_tag* | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*] | [Equations = *Equations. . .*])

Scope

Aria Material

Summary

User field user expression

Description

Provides an expression with the name given by the user_tag parameter using the provided user field. The provided user tag should not include operator, subindex, phase, component, or material phase prefixes or suffixes. See *General Naming Convention* for more detail about those.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
user_tag	"string"	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–
Equations	string. . .	–

User_Function

Syntax

User Expression ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) = User_Function (User_Tag = *user_tag* | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*] | [Equations = *Equations. . .*])

Scope

Aria Material

Summary

User function user expression

Description

Provides an expression with the name given by the user_tag parameter using the provided user function. The provided user tag should not include operator, subindex, phase, component, or material phase prefixes or suffixes. See *General Naming Convention* for more detail about those.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
user_tag	“string”	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–
Equations	string. . .	–

User_Plugin

Syntax

User Expression ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) = User_Plugin (User_Tag = *user_tag* | Name = *Name* | [*plugin_parameters*] | [Equations = *Equations. . .*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
user_tag	"string"	–
Name	"string"	–
plugin_parameters	"string"	–
Equations	string. . .	–

Vector_String_Function

Syntax

User Expression ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) = Vector_String_Function (User_Tag = *user_tag* | F_X = *f_x* | F_Y = *f_y* | [F_Z = *f_z*] | [Equations = *Equations. . .*])

Scope

Aria Material

Summary

Vector string function user expression

Description

Provides an expression with the name given by the user_tag parameter using the provided string function. The provided user tag should not include operator, subindex, phase, component, or material phase prefixes or suffixes. See *General Naming Convention* for more detail about those.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
user_tag	"string"	–
f_x	"string"	–
f_y	"string"	–
f_z	"string"	–
Equations	string. . .	–

Vector_Sum

Syntax

User Expression ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) = Vector_Sum (User_Tag = *user_tag* | Contributions = *contributions* | [Equations = *Equations. . .*])

Scope

Aria Material

Summary

Vector sum

Description

Summation of a list of vector and/or scalar contributions. Scalar sums are applied to all vector components

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
user_tag	"string"	–
contributions	"string"	–
Equations	string. . .	–

V

Valence

Constant

Syntax

Valence [*{of} SpeciesName*] = Constant *{value | z}* = *z*

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
z	real	–

Copied

Syntax

Valence [*{of} SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	“string”	–

Exponential

Syntax

Valence [*{of}* *SpeciesName*] = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Scope

Aria Material

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Valence [*{of}* *SpeciesName*] = Global (Global_Name = *global_name* | [Component = *component*])

Scope

Aria Material

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Valence [*{of}* *SpeciesName*] = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Valence [*{of}* *SpeciesName*] = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Valence [*{of} SpeciesName*] = Scalar_String_Function F = *f*

Scope

Aria Material

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	"string"	–

User_Field

Syntax

Valence [*{of}* *SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Valence [*{of}* *SpeciesName*] = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Scope

Aria Material

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Valence [*{of}* *SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	“string”	–
plugin_parameters	“string”	–

Vector Potential

Constant

Syntax

Vector Potential [*{of}* *SpeciesName*] = Constant ([X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
x	real	0
y	real	0
z	real	0

Copied

Syntax

Vector Potential [*{of}* *SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	"string"	–

User_Field

Syntax

Vector Potential [*{of}* *SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Vector from User Field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Vector Potential [*{of}* *SpeciesName*] = User_Function ([Name_X = *name_x*] | [Name_Y = *name_y*] | [Name_Z = *name_z*] | X = *x* | [Multiplier = *multiplier*])

Scope

Aria Material

Summary

Value from a Vector User Function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name_x	"string"	–
name_y	"string"	–
name_z	"string"	–
x	"string"	–
multiplier	real	1

User_Plugin

Syntax

Vector Potential [*{of}* *SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	"string"	–
plugin_parameters	"string"	–

Vector_String_Function

Syntax

Vector Potential [*{of} SpeciesName*] = Vector_String_Function (F_X = *f_x* | F_Y = *f_y* | [F_Z = *f_z*])

Scope

Aria Material

Summary

Value from a string function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f_x	"string"	–
f_y	"string"	–
f_z	"string"	–

Vector Potential X

Constant

Syntax

Vector Potential X [*{of} SpeciesName*] = Constant Value = *value*

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
value	real	–

Copied

Syntax

Vector Potential X [*{of}* *SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	“string”	–

Exponential

Syntax

Vector Potential X [*{of}* *SpeciesName*] = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Scope

Aria Material

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	“string”	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Vector Potential X [*{of}* *SpeciesName*] = Global (Global_Name = *global_name* | [Component = *component*])

Scope

Aria Material

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Vector Potential X [*{of}* *SpeciesName*] = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
vars	"string"1["string"2["string"3["string"4["string"5]]]]	–
i0	"string"	–
i1	"string"	–
i2	"string"	–
x	"string"	–
y	"string"	–
z	"string"	–

Polynomial

Syntax

Vector Potential X [*{of}* *SpeciesName*] = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Vector Potential X [*{of}* *SpeciesName*] = Scalar_String_Function F = *f*

Scope

Aria Material

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be

a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	"string"	–

User_Field

Syntax

Vector Potential X [*{of}* *SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Vector Potential X [*{of}* *SpeciesName*] = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Scope

Aria Material

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Vector Potential X [*{of} SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	“string”	–
plugin_parameters	“string”	–

Vector Potential Y

Constant

Syntax

Vector Potential Y [*{of} SpeciesName*] = Constant Value = *value*

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
value	real	–

Copied

Syntax

Vector Potential Y [*{of}* *SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	“string”	–

Exponential

Syntax

Vector Potential Y [*{of}* *SpeciesName*] = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Scope

Aria Material

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Vector Potential Y [*{of} SpeciesName*] = Global (Global_Name = *global_name* | [Component = *component*])

Scope

Aria Material

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Vector Potential Y [*{of} SpeciesName*] = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Vector Potential Y [*{of}* *SpeciesName*] = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Vector Potential Y [*{of} SpeciesName*] = Scalar_String_Function F = *f*

Scope

Aria Material

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	"string"	–

User_Field

Syntax

Vector Potential Y [*{of}* *SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Vector Potential Y [*{of}* *SpeciesName*] = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Scope

Aria Material

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Vector Potential Y [*{of}* *SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	“string”	–
plugin_parameters	“string”	–

Vector Potential Z

Constant

Syntax

Vector Potential Z [*{of}* *SpeciesName*] = Constant Value = *value*

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
value	real	–

Copied

Syntax

Vector Potential Z [*{of}* *SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	“string”	–

Exponential

Syntax

Vector Potential Z [*{of}* *SpeciesName*] = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Scope

Aria Material

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Vector Potential Z [*{of} SpeciesName*] = Global (Global_Name = *global_name* | [Component = *component*])

Scope

Aria Material

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Vector Potential Z [*{of} SpeciesName*] = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Vector Potential Z [*{of}* *SpeciesName*] = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Vector Potential Z [*{of} SpeciesName*] = Scalar_String_Function F = *f*

Scope

Aria Material

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	"string"	–

User_Field

Syntax

Vector Potential Z [*{of} SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Vector Potential Z [*{of} SpeciesName*] = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Scope

Aria Material

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Vector Potential Z [*{of} SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	“string”	–
plugin_parameters	“string”	–

Velocity

Constant

Syntax

Velocity [*{of} SpeciesName*] = Constant ([X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
x	real	0
y	real	0
z	real	0

Copied

Syntax

Velocity [*{of}* *SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	“string”	–

Darcy

Syntax

Velocity [*{of}* *SpeciesName*] = Darcy ([Gx = *gx*] | [Gy = *gy*] | [Gz = *gz*] | [Relative_To = *relative_to*])

Scope

Aria Material

Summary

Provides the Darcy velocity model.

Description

For phase p, the velocity provided is

$$\mathbf{u}_p = \phi S_p \mathbf{u}_{matrix} - \frac{k_{r,p}}{\mu_p} \mathbf{K}_s \left(\nabla P_p - \rho_p \mathbf{g} \right),$$

where $\mathbf{u}_{matrix}$ is the optional matrix velocity as specified in the `relative to` parameter, ϕ is the porosity, S_p is the optional phase saturation, $k_{r,p}$ is the relative permeability of the phase, μ_p is the dynamic viscosity of the phase, $\mathbf{K}_s$

is the intrinsic permeability tensor, P_p is the phase pressure, ρ_p is the density of the phase, and **g** is the gravitational acceleration vector.

If the optional phase saturation S_p is provided, the pore volume is assumed to be occupied by multiple phases (e.g., liquid / gas) in accordance to their saturations, otherwise it is occupied fully by a single phase. Note, the sum of phase saturation should equal to one i.e., individual phase volumes should sum to the total pore volume.

Additionally, if the matrix is deforming, the optional `relative to` parameter should specify which phase should be used as the matrix velocity. For instance, `relative to = solid_phase` would ensure the velocity due to the pressure gradient is relative to the deforming solid matrix.

Lastly, note the Darcy velocity is the superficial velocity and is related to the pore scale velocity through the product of porosity and optional phase saturation.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
gx	real	0
gy	real	0
gz	real	0
relative_to	"string"	–

Darcy_Solvent

Syntax

Velocity [*{of}* *SpeciesName*] = Darcy_Solvent ([Gx = *gx*] | [Gy = *gy*] | [Gz = *gz*] | [Relative_To = *relative_to*])

Scope

Aria Material

Summary

This model is intended for problems where the porous species equation is being solved for pressure with a solvent species concentration as the mass term in order to solve for the velocity of that solvent species.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
gx	real	0
gy	real	0
gz	real	0
relative_to	“string”	–

From_Mesh_Displacement

Syntax

Velocity [*{of}* *SpeciesName*] = From_Mesh_Displacement ()

Scope

Aria Material

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–

Melting_Capillary_Darcy

Syntax

Velocity [*{of}* *SpeciesName*] = Melting_Capillary_Darcy ([Melt_Dt = *melt_DT*] | [Melt_Time = *melt_time*] | [Gx = *gx*] | [Gy = *gy*] | [Gz = *gz*])

Scope

Aria Material

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
melt_DT	real	5
melt_time	real	1.798e+308
gx	real	0
gy	real	0
gz	real	0

Schloegl

Syntax

Velocity [*{of}* *SpeciesName*] = Schloegl (K_Phi = *k_phi* | F = *f* | [Gx = *gx*] | [Gy = *gy*] | [Gz = *gz*] | [Proton_Name = *proton_name*])

Scope

Aria Material

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
k_phi	real	–
f	real	–
gx	real	0
gy	real	0
gz	real	0
proton_name	"string"	Hp

User_Field

Syntax

Velocity [*{of}* *SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Vector from User Field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Velocity [*{of}* *SpeciesName*] = User_Function ([Name_X = *name_x*] | [Name_Y = *name_y*] | [Name_Z = *name_z*] | X = *x* | [Multiplier = *multiplier*])

Scope

Aria Material

Summary

Value from a Vector User Function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name_x	"string"	–
name_y	"string"	–
name_z	"string"	–
x	"string"	–
multiplier	real	1

User_Plugin

Syntax

Velocity [*{of}* *SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	"string"	–
plugin_parameters	"string"	–

Vector_String_Function

Syntax

Velocity [*{of} SpeciesName*] = Vector_String_Function (F_X = *f_x* | F_Y = *f_y* | [F_Z = *f_z*])

Scope

Aria Material

Summary

Value from a string function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f_x	"string"	–
f_y	"string"	–
f_z	"string"	–

Velocity Flux

Constant

Syntax

Velocity Flux [*{of} SpeciesName*] = Constant Value = *value*

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
value	real	–

Copied

Syntax

Velocity Flux [*{of}* *SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	"string"	–

Exponential

Syntax

Velocity Flux [*{of}* *SpeciesName*] = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Scope

Aria Material

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Velocity Flux [*{of} SpeciesName*] = Global (Global_Name = *global_name* | [Component = *component*])

Scope

Aria Material

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Velocity Flux [*{of} SpeciesName*] = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
vars	"string"1["string"2["string"3["string"4["string"5]]]]	–
i0	"string"	–
i1	"string"	–
i2	"string"	–
x	"string"	–
y	"string"	–
z	"string"	–

Polynomial

Syntax

Velocity Flux [*{of}* *SpeciesName*] = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Velocity Flux [*{of}* *SpeciesName*] = Scalar_String_Function F = *f*

Scope

Aria Material

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be

a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	"string"	–

User_Field

Syntax

Velocity Flux [*{of} SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Velocity Flux [*{of} SpeciesName*] = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Scope

Aria Material

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1
toggle	"string"	–

User_Plugin

Syntax

Velocity Flux [*{of} SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	"string"	–
plugin_parameters	"string"	–

Velocity X

Constant

Syntax

Velocity X [*{of} SpeciesName*] = Constant Value = *value*

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
value	real	–

Copied

Syntax

Velocity X [*{of}* *SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	“string”	–

Exponential

Syntax

Velocity X [*{of}* *SpeciesName*] = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Scope

Aria Material

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Velocity X [*{of} SpeciesName*] = Global (Global_Name = *global_name* | [Component = *component*])

Scope

Aria Material

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Velocity X [*{of} SpeciesName*] = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Velocity X [*{of}* *SpeciesName*] = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Velocity X [*{of}* *SpeciesName*] = Scalar_String_Function F = *f*

Scope

Aria Material

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	"string"	–

User_Field

Syntax

Velocity X [*{of}* *SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Velocity X [*{of}* *SpeciesName*] = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Scope

Aria Material

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Velocity X [*{of} SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	“string”	–
plugin_parameters	“string”	–

Velocity Y

Constant

Syntax

Velocity Y [*{of} SpeciesName*] = Constant Value = *value*

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
value	real	–

Copied

Syntax

Velocity Y [*{of}* *SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	“string”	–

Exponential

Syntax

Velocity Y [*{of}* *SpeciesName*] = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Scope

Aria Material

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Velocity Y [*{of} SpeciesName*] = Global (Global_Name = *global_name* | [Component = *component*])

Scope

Aria Material

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Velocity Y [*{of} SpeciesName*] = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Velocity Y [*{of}* *SpeciesName*] = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Velocity Y [*{of}* *SpeciesName*] = Scalar_String_Function F = *f*

Scope

Aria Material

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	"string"	–

User_Field

Syntax

Velocity Y [*{of}* *SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Velocity Y [*{of}* *SpeciesName*] = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Scope

Aria Material

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Velocity Y [*{of} SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	“string”	–
plugin_parameters	“string”	–

Velocity Z

Constant

Syntax

Velocity Z [*{of} SpeciesName*] = Constant Value = *value*

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
value	real	–

Copied

Syntax

Velocity Z [*{of}* *SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	“string”	–

Exponential

Syntax

Velocity Z [*{of}* *SpeciesName*] = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Scope

Aria Material

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Velocity Z [*{of}* *SpeciesName*] = Global (Global_Name = *global_name* | [Component = *component*])

Scope

Aria Material

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Velocity Z [*{of}* *SpeciesName*] = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Velocity Z [*{of}* *SpeciesName*] = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Velocity Z [*{of}* *SpeciesName*] = Scalar_String_Function F = *f*

Scope

Aria Material

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	"string"	–

User_Field

Syntax

Velocity Z [*{of} SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Velocity Z [*{of} SpeciesName*] = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Scope

Aria Material

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Velocity Z [*{of}* *SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	“string”	–
plugin_parameters	“string”	–

Venting Volumetric Flow Rate

Constant

Syntax

Venting Volumetric Flow Rate [*{of}* *SpeciesName*] = Constant Value = *value*

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
value	real	–

Copied

Syntax

Venting Volumetric Flow Rate [*{of}* *SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	“string”	–

Exponential

Syntax

Venting Volumetric Flow Rate [*{of}* *SpeciesName*] = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Scope

Aria Material

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Venting Volumetric Flow Rate [*{of}* *SpeciesName*] = Global (Global_Name = *global_name* | [Component = *component*])

Scope

Aria Material

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
global_name	"string"	–
component	integer	0

K_Factor

Syntax

Venting Volumetric Flow Rate [*{of}* *SpeciesName*] = K_Factor [Flow_Rate_Limiter = *flow_rate_limiter*]

Scope

Aria Material

Summary

Volumetric flow rate, Q, calculated as $Q = K\sqrt{\frac{P-P_{ambient}}{\rho}}$. Absent any other sources, the maximum flow rate is defined based on the mass loss to achieve pressure equilibrium and the fluid time step. If the optional flow rate limiter is specified, the flow rate is limited by the product of the limiter factor and the maximum flow rate without any other sources.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
flow_rate_limiter	real	0

K_Factor_With_Choking

Syntax

Venting Volumetric Flow Rate [*{of}* *SpeciesName*] = K_Factor_With_Choking (Critical_Pressure_Ratio = *critical_pressure_ratio* | [Flow_Rate_Limiter = *flow_rate_limiter*])

Scope

Aria Material

Summary

Volumetric flow rate, Q, calculated as $Q = K\sqrt{\frac{P-P_{ambient}}{\rho}}$ for $\frac{P}{P_{ambient}} < R_p$, where R_p is the critical_pressure_ratio. Above R_p the flow rate is calculated as $Q = K\sqrt{\frac{P_{high}(1-1/R_p)}{\rho}}$. Absent any other sources, the maximum flow rate is defined based on the mass loss to achieve pressure equilibrium and the fluid time step. If the optional flow rate limiter is specified, the flow rate is limited by the product of the limiter factor and the maximum flow rate without any other sources.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
critical_pressure_ratio	real	–
flow_rate_limiter	real	0

Nd_Table

Syntax

Venting Volumetric Flow Rate [*{of}* *SpeciesName*] = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Venting Volumetric Flow Rate [*{of}* *SpeciesName*] = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Venting Volumetric Flow Rate [*{of}* *SpeciesName*] = Scalar_String_Function F = *f*

Scope

Aria Material

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	"string"	–

User_Field

Syntax

Venting Volumetric Flow Rate [*{of}* *SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Venting Volumetric Flow Rate [*{of}* *SpeciesName*] = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Scope

Aria Material

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1
toggle	"string"	–

User_Plugin

Syntax

Venting Volumetric Flow Rate [*{of}* *SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	"string"	–
plugin_parameters	"string"	–

Viscosity

Arrhenius

Syntax

Viscosity [*{of}* *SpeciesName*] = Arrhenius (Mu0 = *mu0* | E = *e* | [Mu_Max = *mu_max*])

Scope

Aria Material

Summary

A viscosity model dependent on the temperature T

$$\mu(T) = \mu_0 e^{-E/T},$$

where μ_0 and E are model parameters.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
mu0	real	–
e	real	–
mu_max	real	1.798e+308

Arrhenius_Carreau

Syntax

Viscosity [*{of}* *SpeciesName*] = Arrhenius_Carreau (Mu_Zero = *mu_zero* | [Mu_Inf = *mu_inf*] | N = *n* | [A = *a*] | [Lambda = *lambda*] | K = *k* | T_Ref = *t_ref* | [Skip_Sensitivities = *skip_sensitivities*] | [Sensitivity_Scaling = *sensitivity_scaling*] | [Ramp_Sensitivities_Over_Time = *ramp_sensitivities_over_time*])

Scope

Aria Material

Summary

This viscosity model is a function of the shear rate (Carreau model) and temperature (Arrhenius model). The two contributions are combined multiplicatively.

$$\mu\left(\dot{\gamma},T\right)=\left[\mu_{\infty}+\left(\mu_{\circ}-\mu_{\infty}\right)\left(1+\left(\lambda\dot{\gamma}e^{\frac{K}{T}-\frac{K}{T_{ref}}}\right)^{a}\right)^{\frac{n-1}{a}}\right],$$

where μ_{∞} is the infinite shear viscosity (defaults to zero), $\mu_{\circ}$ is the zero shear viscosity, n and a (defaults to 2) are model parameters, $\dot{\gamma}$ is the shear rate, λ is a time constant (defaults to 1), T_{ref} is a reference temperature and K is an Arrhenius constant.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
mu_zero	real	–
mu_inf	real	0
n	real	–
a	real	2
lambda	real	1
k	real	–
t_ref	real	–
skip_sensitivities	integer	–
sensitivity_scaling	real	1
ramp_sensitivities_over_time	real	-1

Bingham_Wlf

Syntax

Viscosity [*{of}* *SpeciesName*] = Bingham_Wlf (Mu_Zero = *mu_zero* | Mu_Inf = *mu_inf* | F = *f* | N = *n* | A = *a* | Lambda = *lambda* | Tau_Y = *tau_y* | [Skip_Sensitivities = *skip_sensitivities*])

Scope

Aria Material

Summary

A viscosity model dependent on the shear rate $\dot{\gamma}$

$$\mu(\dot{\gamma}) = \mu_\infty + \left(\mu_\circ - \mu_\infty + \tau_y \frac{1 - e^{-\dot{\gamma}F}}{\dot{\gamma}}\right)\left(1 + (\lambda\dot{\gamma})^a\right)^{\frac{n-1}{a}},$$

where μ_∞ is the infinite shear viscosity, $\mu_\circ$ is the zero shear viscosity, λ is a time constant, τ_y is the yield stress, and F, a and n are additional model parameters.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
mu_zero	real	–
mu_inf	real	–
f	real	–
n	real	–
a	real	–
lambda	real	–
tau_y	real	–
skip_sensitivities	integer	0

Bingham_Wlft

Syntax

Viscosity [*{of}* *SpeciesName*] = Bingham_Wlft (Mu_Zero = *mu_zero* | Mu_Inf = *mu_inf* | F = *f* | N = *n* | A = *a* | Lambda = *lambda* | Tau_Y = *tau_y* | T_Ref = *t_ref* | C1 = *c1* | C2 = *c2* | [Skip_Sensitivities = *skip_sensitivities*])

Scope

Aria Material

Summary

A viscosity model dependent on the shear rate $\dot{\gamma}$ and temperature T

$$\mu(\dot{\gamma}, T) = a_T \left(\mu_\infty + \left(\mu_\circ - \mu_\infty + \tau_y \frac{1 - e^{-a_T \dot{\gamma} F}}{a_T \dot{\gamma}} \right) \left(1 + (a_T \lambda \dot{\gamma})^a \right)^{\frac{n-1}{a}} \right),$$

where μ_∞ is the infinite shear viscosity, $\mu_\circ$ is the zero shear viscosity, τ_y is the yield stress, λ is the time constant, and where

$$a_T = e^{\frac{c_1 (T_\circ - T)}{c_2 + T - T_\circ}},$$

with $T_\circ$ the reference temperature. Additional model parameters are a, n, c_1, and c_2.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
mu_zero	real	–
mu_inf	real	–
f	real	–
n	real	–
a	real	–
lambda	real	–
tau_y	real	–
t_ref	real	–
c1	real	–
c2	real	–
skip_sensitivities	integer	–

Cantera

Syntax

Viscosity [*{of}* *SpeciesName*] = Cantera [Poffset = *poffset*]

Scope

Aria Material

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
poffset	real	0

Carreau

Syntax

Viscosity [*{of}* *SpeciesName*] = Carreau (Mu_Zero = *mu_zero* | [Mu_Inf = *mu_inf*] | N = *n* | [A = *a*] | Lambda = *lambda* | [Skip_Sensitivities = *skip_sensitivities*])

Scope

Aria Material

Summary

A viscosity model dependent on the shear rate $\dot{\gamma}$

$$\frac{\mu(\dot{\gamma}) - \mu_\infty}{\mu_\circ - \mu_\infty} = (1 + (\lambda\dot{\gamma})^a)^{\frac{n-1}{a}},$$

where μ_∞ is the infinite shear viscosity (defaults to zero), $\mu_\circ$ is the zero shear viscosity, n and a (defaults to 2) are model parameters, and λ is a time constant.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
mu_zero	real	–
mu_inf	real	0
n	real	–
a	real	2
lambda	real	–
skip_sensitivities	integer	0

Carreau_T

Syntax

Viscosity [*{of}* *SpeciesName*] = Carreau_T (Mu_Zero = *mu_zero* | [Mu_Inf = *mu_inf*] | N = *n* | [A = *a*] | K = *k* | [Skip_Sensitivities = *skip_sensitivities*])

Scope

Aria Material

Summary

A viscosity model dependent on the shear rate $\dot{\gamma}$ and temperature T

$$\frac{\mu(\dot{\gamma},T)-\mu_\infty}{\mu_\circ-\mu_\infty}=\left(1+\left(e^{k/T}\dot{\gamma}\right)^a\right)^{\frac{n-1}{a}},$$

where μ_∞ is the infinite shear viscosity (defaults to zero), $\mu_\circ$ is the zero shear viscosity , and n, a (defaults to 2), and k are model parameters. The quantity $e^{k/T}$ is a temperature dependent time scale replacing the time scale λ in the Carreau model.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
mu_zero	real	–
mu_inf	real	0
n	real	–
a	real	2
k	real	–
skip_sensitivities	integer	–

Casson

Syntax

Viscosity [*{of}* *SpeciesName*] = Casson (Mu_Zero = *mu_zero* | M = *m* | Shear_Reference = *shear_reference* | Yield_Stress = *yield_stress* | [Skip_Sens_Iteration_Count = *skip_sens_iteration_count*])

Scope

Aria Material

Summary

A viscosity model dependent on the shear rate $\dot{\gamma}$

$$\mu\left(\dot{\gamma}\right)=\left[\sqrt{\mu_\circ}+\sqrt{\tau_y/\dot{\gamma}}\left(1-e^{-\sqrt{m\dot{\gamma}/\dot{\gamma}_{ref}}}\right)\right]^2,$$

where μ_∞ is the infinite shear viscosity, $\mu_\circ$ is the zero shear viscosity, τ_y is the yield stress, $\dot{\gamma}_{ref}$ is a reference shear rate, and m is a model parameter.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
mu_zero	real	–
m	real	–
shear_reference	real	–
yield_stress	real	–
skip_sens_iteration_count	integer	0

Clsm

Syntax

Viscosity [*{of}* *SpeciesName*] = Clsm (Primaryproperty = *primaryProperty* | Secondaryproperty = *secondaryProperty*)

Scope

Aria Material

Summary

CLSM viscosity

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
primaryProperty	real	–
secondaryProperty	real	–

Constant

Syntax

Viscosity [*{of}* *SpeciesName*] = Constant *{mu | value}* = *mu*

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
mu	real	–

Copied

Syntax

Viscosity [*{of}* *SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	“string”	–

Curing_Epoxy

Syntax

Viscosity [*{of}* *SpeciesName*] = Curing_Epoxy (Mu_A = *mu_a* | T_Mu = *t_mu* | N_Mu = *n_mu* | E_Mu = *e_mu* | R = *r* | Ksi_C = *ksi_c* | M_Ksi_C = *m_ksi_c* | [Epoxy_Subindex = *epoxy_subindex*] | [Epoxy_Species_Name = *epoxy_species_name*] | [Epoxy_Species_Phase = *epoxy_species_phase*])

Scope

Aria Material

Summary

For a curing epoxy with extent of reaction ξ the viscosity is given by

$$\mu\left(\xi, T\right) = \mu_a \left(1 - \frac{T}{T_\mu}\right)\left(1 - \left(\frac{\xi}{\xi_c}\right)^m\right)^{n_\mu} e^{E_\mu / RT}$$

where T is the temperature and the default model parameters are $\mu_a = 800.93$, $T_\mu = 364.059$, $n_\mu = -4/3$, $E_\mu = 0.0$, $R = 1.987$, and $\xi_c = 0.45$. **NOTE:** The extent of reaction field is assumed to be a `SPECIES` field with the subindex provided by the `EPOXY_SUBINDEX` parameter.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
mu_a	real	–
t_mu	real	–
n_mu	real	–
e_mu	real	–
r	real	–
ksi_c	real	–
m_ksi_c	real	–
epoxy_subindex	integer	-1
epoxy_species_name	"string"	–
epoxy_species_phase	"string"	–

Curing_Foam

Syntax

Viscosity [*{of}* *SpeciesName*] = Curing_Foam (Mu_A = *mu_a* | T_Mu = *t_mu* | N_Mu = *n_mu* | E_Mu = *e_mu* | R = *r* | Ksi_C = *ksi_c* | M_Ksi_C = *m_ksi_c* | Extent_Subindex = *extent_subindex*)

Scope

Aria Material

Summary

For a curing foam with gas volume fraction ϕ_g and extent of reaction ξ the viscosity is given by

$$\mu\left(\xi,\phi_g,T\right)=\mu_a\left(1-\frac{T}{T_\mu}\right)\left(1-\left(\frac{\xi}{\xi_c}\right)^m\right)^{n_\mu}e^{E_\mu/RT}e^{\phi_g/\left(1-\phi_g\right)}$$

where T is the temperature and the default model parameters are $\mu_a = 800.93$, $T_\mu = 364.059$, $n_\mu = -4/3$, $E_\mu = 0.0$, $R = 1.987$, and $\xi_c = 0.45$. **NOTE:** The gas volume fraction is assumed to be a `SPECIES` field with the subindex provided by the `VFRAC_SUBINDEX` parameter. Likewise, the extent of reaction field is assumed to be a `SPECIES` field with the subindex provided by the `EXTENT_SUBINDEX` parameter.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
mu_a	real	–
t_mu	real	–
n_mu	real	–
e_mu	real	–
r	real	–
ksi_c	real	–
m_ksi_c	real	–
extent_subindex	integer	–

Exponential

Syntax

Viscosity [*{of}* *SpeciesName*] = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Scope

Aria Material

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Viscosity [*{of}* *SpeciesName*] = Global (Global_Name = *global_name* | [Component = *component*])

Scope

Aria Material

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
global_name	“string”	–
component	integer	0

Interpolated_Phase_Average

Syntax

Viscosity [*{of}* *SpeciesName*] = Interpolated_Phase_Average ([Subindex_A = *subindex_a*] | [Subindex_B = *subindex_b*])

Scope

Aria Material

Summary

Phase-averaged values

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
subindex_a	integer	-1
subindex_b	integer	-1

Keyes

Syntax

Viscosity [*{of}* *SpeciesName*] = Keyes (A0 = *a0* | A = *a* | A1 = *a1*)

Scope

Aria Material

Summary

A viscosity model dependent on the temperature T

$$\mu(T) = ka_0 \frac{T^{3/2}}{T + a10^{-a_1/T}},$$

where $k = 1 \times 10^{-6}$ and where a_0, a, a_1 are model parameters.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
a0	real	–
a	real	–
a1	real	–

Krieger

Syntax

Viscosity [*{of}* *SpeciesName*] = Krieger (Beta = *beta* | Phi_Max = *phi_max* | Mu_S = *mu_s* | [Use_Fd = *use_fd*])

Scope

Aria Material

Summary

A viscosity model [56] dependent on the suspension concentration ϕ

$$\mu\left(\phi\right) = \mu_s \left(1 - \frac{\phi}{\phi_{max}}\right)^{\beta},$$

where β is the Krieger exponent, ϕ_{max} is the maximum suspension concentration and μ_s is the solvent viscosity.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
beta	real	–
phi_max	real	–
mu_s	real	–
use_fd	integer	0

Mass_Average

Syntax

Viscosity [*{of}* *SpeciesName*] = Mass_Average ()

Scope

Aria Material

Summary

Viscosity computed from mass average of species viscosities

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–

Mixture_Fraction

Syntax

Viscosity [*{of}* *SpeciesName*] = Mixture_Fraction (Primaryproperty = *primaryProperty* \| Secondaryproperty = *secondaryProperty*)

Scope

Aria Material

Summary

Viscosity model dependent on the mixture fraction f

$$\mu(f) = f\mu_1 + (1 - f)\,\mu_2,$$

where μ_1 and μ_2 are viscosities corresponding to $f = 1$ and $f = 0$ respectively.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
primaryProperty	real	–
secondaryProperty	real	–

Mixture_Fraction_Turbulent

Syntax

Viscosity [*{of}* *SpeciesName*] = Mixture_Fraction_Turbulent (Primaryproperty = *primaryProperty* | Secondaryproperty = *secondaryProperty*)

Scope

Aria Material

Summary

Turbulent viscosity model dependent on the mixture fraction f

$$\mu(f) = f\mu_1 + (1 - f)\,\mu_2,$$

where μ_1 and μ_2 are viscosities corresponding to $f = 1$ and $f = 0$ respectively.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
primaryProperty	real	–
secondaryProperty	real	–

Morris_Boulay

Syntax

Viscosity [*{of}* *SpeciesName*] = Morris_Boulay (K_S = *K_s* | Phi_Max = *phi_max* | Phi_Tol = *phi_tol* | Mu_S = *mu_s*)

Scope

Aria Material

Summary

A viscosity model for noncolloidal suspensions

$$\mu(\phi) = \mu_s \left(1 + 2.5\phi_{max}\left(1 - \bar{\phi}\right)^{-1} + K_s\bar{\phi}^2\left(1 - \bar{\phi}\right)^{-2}\right),$$

where ϕ is the volume fraction, $\bar{\phi} = \phi/\phi_{max}$ is the volume fraction normalized by the max volume fraction, μ_s is the suspending fluid viscosity, and K_s is a model parameter. **NOTE:** $\bar{\phi}$ is bounded such that

$$0 \le \bar{\phi} \le \frac{\phi_{tol}}{\phi_{max}},$$

where ϕ_{tol} is the max attainable volume fraction.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
K_s	real	–
phi_max	real	–
phi_tol	real	–
mu_s	real	–

Nd_Table

Syntax

Viscosity [*{of}* *SpeciesName*] = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Phase_Average

Syntax

Viscosity [*{of}* *SpeciesName*] = Phase_Average ([Subindex_A = *subindex_a*] | [Subindex_B = *subindex_b*])

Scope

Aria Material

Summary

Phase-averaged values

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
subindex_a	integer	-1
subindex_b	integer	-1

Polynomial

Syntax

Viscosity [*{of}* *SpeciesName*] = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Power_Law

Syntax

Viscosity [*{of}* *SpeciesName*] = Power_Law ([K = *k*] | N = *n*)

Scope

Aria Material

Summary

A viscosity model dependent on the shear rate $\dot{\gamma}$

$$\mu\left(\dot{\gamma}\right) = K\dot{\gamma}^{N-1},$$

where K is a model parameter, and N is the power law exponent.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
k	real	1
n	real	–

Ramaccioti

Syntax

Viscosity [*{of}* *SpeciesName*] = Ramaccioti ([Interpolated = *interpolated*] | T_Liq = *T_liq* | T_Sol = *T_sol* | C = *c* | Mu_Liq = *mu_liq*)

Scope

Aria Material

Summary

A viscosity model used for corium melts

$$\mu\left(T\right) = \mu_{liq} e^{2.5 C f_s},$$

where $f_s = \min(\bar{f_s}, 1)$ with $\bar{f_s}$ defined as

$$\bar{f_s} = \begin{cases} \dfrac{T_{liq} - T}{T_{liq} - T_{sol}} & T < T_{liq} \\ 0 & T \geq T_{liq}, \end{cases}$$

where T is the temperature, μ_{liq} is the viscosity of the liquid, T_{liq} is the liquidus temperature, T_{sol} is the solidus temperature and $4 \leq C \leq 8$ is a model parameter.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
interpolated	integer	0
T_liq	real	–
T_sol	real	–
c	real	–
mu_liq	real	–

Scalar_String_Function

Syntax

Viscosity [*{of}* *SpeciesName*] = Scalar_String_Function F = *f*

Scope

Aria Material

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name.

Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	"string"	–

Species_Average

Syntax

Viscosity [*{of}* *SpeciesName*] = Species_Average ()

Scope

Aria Material

Summary

Viscosity computed from mass average of species viscosities

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–

Sutherland

Syntax

Viscosity [*{of}* *SpeciesName*] = Sutherland (Tref = *tref* | C = *c*)

Scope

Aria Material

Summary

A viscosity model dependent on temperature T

$$\mu\left(T\right) = \frac{cT^{1.5}}{T + T_{ref}},$$

where T_{ref} and c are model parameters.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
tref	real	–
c	real	–

Thermal

Syntax

Viscosity [*{of} SpeciesName*] = Thermal ([A = *a*] | [B = *b*] | [C = *c*] | [D = *d*])

Scope

Aria Material

Summary

A viscosity model dependent on the temperature T

$$\mu(T) = A + BT + CT^2 + DT^3,$$

where A, B, C, D are coefficients of a fitted cubic polynomial.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
a	real	0
b	real	0
c	real	0
d	real	0

User_Field

Syntax

Viscosity [*{of} SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
scaling	real	1
global_var	“string”	–
time_function	“string”	–

User_Function

Syntax

Viscosity [*{of}* *SpeciesName*] = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Scope

Aria Material

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Viscosity [*{of}* *SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	"string"	–
plugin_parameters	"string"	–

Volume_Average

Syntax

Viscosity [*{of}* *SpeciesName*] = Volume_Average ()

Scope

Aria Material

Summary

Viscosity computed from volume average of species viscosities, over a material phase

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–

Weld

Syntax

Viscosity [*{of}* *SpeciesName*] = Weld ([Interpolated = *interpolated*] | [Beta = *beta*] | C0 = *c0* | C1 = *c1* | C2 = *c2* | C3 = *c3* | T_Liq = *T_liq* | T_90 = *t_90* | T_Max = *T_max*)

Scope

Aria Material

Summary

This is an empirical model that emulates the melting of a solid metal during the laser welding process.

$$\mu\left(T\right) = \begin{cases} \mu_{90} + \left(\mu_{liq} - \mu_{90}\right)\left(\dfrac{T - T_{90}}{T_{liq} - T_{90}}\right) & T < T_{liq} \\ c_0 + c_1\hat{T} + c_2\hat{T}^2 + c_3\hat{T}^3 & T \geq T_{liq}, \end{cases}$$

where μ_{liq} is given by

$$\mu_{liq} = c_0 + c_1 T_{liq} + c_2 T_{liq}^2 + c_3 T_{liq}^3,$$

and where $\mu_{90} = \beta\mu_{liq}$ and $\hat{T} = \min(T, T_{max})$. The default value of β is 10^{11}.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
interpolated	integer	0
beta	real	1e+11
c0	real	–
c1	real	–
c2	real	–
c3	real	–
T_liq	real	–
t_90	real	–
T_max	real	–

Voltage

Constant

Syntax

Voltage [*{of}* *SpeciesName*] = Constant ([Toggle = *toggle*] | Value = *value*)

Scope

Aria Material

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
toggle	“string”	–
value	real	–

Copied

Syntax

Voltage [*{of}* *SpeciesName*] = Copied ([Toggle = *toggle*] | Source = *source*)

Scope

Aria Material

Summary

Copied value from another expression

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
toggle	“string”	–
source	“string”	–

Exponential

Syntax

Voltage [*{of}* *SpeciesName*] = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Scope

Aria Material

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Voltage [*{of} SpeciesName*] = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Scope

Aria Material

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
toggle	"string"	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Voltage [*{of} SpeciesName*] = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Voltage [*{of} SpeciesName*] = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
toggle	"string"	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Voltage [*{of}* *SpeciesName*] = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Scope

Aria Material

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name.

Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
toggle	"string"	–
f	"string"	–

User_Field

Syntax

Voltage [*{of} SpeciesName*] = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
toggle	"string"	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Voltage [*{of} SpeciesName*] = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Scope

Aria Material

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
toggle	“string”	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Voltage [*{of} SpeciesName*] = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is

active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
toggle	“string”	–
Name	“string”	–
plugin_parameters	“string”	–

Volume Fraction

Constant

Syntax

Volume Fraction [*{of}* *SpeciesName*] = Constant Value = *Value*

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Value	real	–

Copied

Syntax

Volume Fraction [*{of}* *SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	“string”	–

Exponential

Syntax

Volume Fraction [*{of}* *SpeciesName*] = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Scope

Aria Material

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

From_Electrode_Object_Old

Syntax

Volume Fraction [*{of}* *SpeciesName*] = From_Electrode_Object_Old (F = *f* | Electrodefile = *electrodeFile* | Kmolconversion = *kmolConversion* | [Jconversion = *JConversion*] | [Meterconversion = *meterConversion*] | [Max_Sub_Timesteps = *max_sub_timesteps*] | [Base_Delta = *base_delta*])

Scope

Aria Material

Summary

Get the solid volume fraction for all species tracked by the Electrode object being used to calculate electrochemical reaction source terms.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	real	–
electrodeFile	"string"	–
kmolConversion	real	–
JConversion	real	1
meterConversion	real	1
max_sub_timesteps	integer	100
base_delta	real	1e-05

From_Mass_Fraction

Syntax

Volume Fraction [*{of}* *SpeciesName*] = From_Mass_Fraction ()

Scope

Aria Material

Summary

Compute the volume fraction from the ratio of species porous density to species solid density

Description

Calculates volume fraction for species i ($V_{f,i}$) using

$$V_{f,i} = \frac{Y_i \rho}{\rho_{s,i}}$$

where ρ is the `DENSITY` expression, Y_i is the mass fraction, and $\rho_{s,i}$ is the `SOLID_DENSITY` for species i.

The overall density material phase to use can be specified with the optional 'Density_Phase_Name' argument, which can be necessary for porous systems depending on how the mass fractions are defined. The provided `Density_Phase_Name` material phase is only applied to ρ.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–

From_Molar_Volume

Syntax

Volume Fraction [*{of}* *SpeciesName*] = From_Molar_Volume ()

Scope

Aria Material

Summary

Compute the volume fraction of a species as the product of it's concentration and partial molar volume.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–

From_Porosity

Syntax

Volume Fraction [*{of}* *SpeciesName*] = From_Porosity ()

Scope

Aria Material

Summary

Compute the volume fraction from a species porosity.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–

Global

Syntax

Volume Fraction [*{of}* *SpeciesName*] = Global (Global_Name = *global_name* | [Component = *component*])

Scope

Aria Material

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
global_name	“string”	–
component	integer	0

Nd_Table

Syntax

Volume Fraction [*{of}* *SpeciesName*] = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Volume Fraction [*{of}* *SpeciesName*] = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Volume Fraction [*{of}* *SpeciesName*] = Scalar_String_Function F = *f*

Scope

Aria Material

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	"string"	–

User_Field

Syntax

Volume Fraction [*{of}* *SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Volume Fraction [*{of}* *SpeciesName*] = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Scope

Aria Material

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Volume Fraction [*{of}* *SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	“string”	–
plugin_parameters	“string”	–

Volume Fraction Gas

Constant

Syntax

Volume Fraction Gas [*{of}* *SpeciesName*] = Constant Value = *value*

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
value	real	–

Copied

Syntax

Volume Fraction Gas [*{of}* *SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	“string”	–

Exponential

Syntax

Volume Fraction Gas [*{of}* *SpeciesName*] = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Scope

Aria Material

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	“string”	–
constant	real	0
multiplier	real	1
exponent	real	–

From_Chemeq_Mass_Fractions

Syntax

Volume Fraction Gas [*{of}* *SpeciesName*] = From_Chemeq_Mass_Fractions ()

Scope

Aria Material

Summary

Compute the gas volume fraction from the mass fractions and bulk densities pulled from ChemEq

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–

From_Chemeq_Species

Syntax

Volume Fraction Gas [*{of}* *SpeciesName*] = From_Chemeq_Species ()

Scope

Aria Material

Summary

Compute the gas volume fraction from the species concentrations pulled from ChemEq

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–

From_Density

Syntax

Volume Fraction Gas [*{of}* *SpeciesName*] = From_Density (Rho_Liquid = *rho_liquid* | Rho_Gas = *rho_gas* | Rho_Initial = *rho_initial*)

Scope

Aria Material

Summary

Compute the gas volume fraction from the gas density

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
rho_liquid	real	–
rho_gas	real	–
rho_initial	real	–

From_Mass_Fractions

Syntax

Volume Fraction Gas [*{of}* *SpeciesName*] = From_Mass_Fractions ()

Scope

Aria Material

Summary

This model has been deprecated and removed, use 'From_Chemeq_Mass_Fractions' instead

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–

From_Porosity

Syntax

Volume Fraction Gas [*{of}* *SpeciesName*] = From_Porosity ()

Scope

Aria Material

Summary

Gas volume fraction from the porosity

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–

From_Reaction_Extent

Syntax

Volume Fraction Gas [*{of}* *SpeciesName*] = From_Reaction_Extent (Species = *species* | Al_1 = *al_1* | Al_2 = *al_2* | V_Liq = *v_liq* | N_I = *n_i* | N_Max = *n_max* | Mw = *mw* | [T_Ref = *t_ref*])

Scope

Aria Material

Summary

Gas volume fraction from reaction extent for foaming applications

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
species	integer	–
al_1	real	–
al_2	real	–
v_liq	real	–
n_i	real	–
n_max	real	–
mw	real	–
t_ref	real	0

From_Species

Syntax

Volume Fraction Gas [*{of}* *SpeciesName*] = From_Species ()

Scope

Aria Material

Summary

This model has been deprecated and removed, use ‘From_Chemeq_Species’ instead

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–

From_Thermal_Expansion

Syntax

Volume Fraction Gas [*{of}* *SpeciesName*] = From_Thermal_Expansion ([Phi0 = *phi0*] | [Rhob0 = *rhob0*] | [Rhoc0 = *rhoc0*] | [Beta0 = *beta0*] | [Beta1 = *beta1*] | [T0 = *t0*])

Scope

Aria Material

Summary

Gas volume fraction from thermal expansion of the condensed phase

Description

This model calculates volume fraction of a porous material where the condensed phase has linear thermal expansion

$$\phi = 1 - \frac{\rho}{\rho_c}$$

The value is clipped to be between 1e-16 and 1, but generally for this model to be relevant the density (ρ) should be less than the condensed density (ρ_c). The condensed density is calculated from

$$\rho_c = \rho_{c,0}\left(1 - \beta\left(T - T_0\right)\right)$$

$$\beta = \beta_0 + \beta_1 T$$

If $\rho_{c,0}$ is not provided, it is calculated from

$$\rho_{c,0} = \frac{\rho_{b,0}}{1 - \phi_0}$$

Note: This model does not account for changes in gas volume fraction due to reactions that convert solids to gasses. It only includes thermal expansion.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
phi0	real	-1
rhob0	real	-1
rhoc0	real	-1
beta0	real	0
beta1	real	0
t0	real	300

Fromfoamtimetemp

Syntax

Volume Fraction Gas [*{of}* *SpeciesName*] = Fromfoamtimetemp (Rho_Liquidfluorinert = *rho_liquidfluorinert* | Rho_Vaporfluorinert = *rho_vaporfluorinert* | Rho_Air = *rho_air* | Massfraction_Air = *massfraction_air* | Rho_Initial = *rho_initial*)

Scope

Aria Material

Summary

Compute the gas volume fraction from the foam time/temperature profile

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
rho_liquidfluorinert	real	–
rho_vaporfluorinert	real	–
rho_air	real	–
massfraction_air	real	–
rho_initial	real	–

Global

Syntax

Volume Fraction Gas [*{of}* *SpeciesName*] = Global (Global_Name = *global_name* | [Component = *component*])

Scope

Aria Material

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
global_name	“string”	–
component	integer	0

Interpolated_Phase_Average

Syntax

Volume Fraction Gas [*{of} SpeciesName*] = Interpolated_Phase_Average ([Subindex_A = *subindex_a*] | [Subindex_B = *subindex_b*])

Scope

Aria Material

Summary

Phase-averaged values

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
subindex_a	integer	-1
subindex_b	integer	-1

Nd_Table

Syntax

Volume Fraction Gas [*{of} SpeciesName*] = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Phase_Average

Syntax

Volume Fraction Gas [*{of}* *SpeciesName*] = Phase_Average ([Subindex_A = *subindex_a*] | [Subindex_B = *subindex_b*])

Scope

Aria Material

Summary

Phase-averaged values

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
subindex_a	integer	-1
subindex_b	integer	-1

Polynomial

Syntax

Volume Fraction Gas [*{of}* *SpeciesName*] = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Volume Fraction Gas [*{of}* *SpeciesName*] = Scalar_String_Function F = *f*

Scope

Aria Material

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	"string"	–

Species

Syntax

Volume Fraction Gas [*{of}* *SpeciesName*] = Species ([Gas_Species_Subindex = *gas_species_subindex*] | [Gas_Species_Name = *gas_species_name*])

Scope

Aria Material

Summary

Compute the gas volume fraction as $C_g * MW_g/rho_g$ where C_g is the molar concentration of the gas, MW_g is the molecular weight, and rho_g is the density of the pure gas at the current T, P. The species to use for C_g and MW_g can be specified either by name or by subindex.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
gas_species_subindex	integer	-1
gas_species_name	"string"	–

User_Field

Syntax

Volume Fraction Gas [*{of}* *SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Volume Fraction Gas [*{of}* *SpeciesName*] = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Scope

Aria Material

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1
toggle	"string"	–

User_Plugin

Syntax

Volume Fraction Gas [*{of}* *SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	“string”	–
plugin_parameters	“string”	–

Volumetric Heat Transfer Coefficient

Constant

Syntax

Volumetric Heat Transfer Coefficient [*{of} SpeciesName*] = Constant *{h | value}* = *h*

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
h	real	–

Copied

Syntax

Volumetric Heat Transfer Coefficient [*{of} SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	“string”	–

Exponential

Syntax

Volumetric Heat Transfer Coefficient [*{of}* *SpeciesName*] = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Scope

Aria Material

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Volumetric Heat Transfer Coefficient [*{of}* *SpeciesName*] = Global (Global_Name = *global_name* | [Component = *component*])

Scope

Aria Material

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Volumetric Heat Transfer Coefficient [*{of} SpeciesName*] = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Volumetric Heat Transfer Coefficient [*{of} SpeciesName*] = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Volumetric Heat Transfer Coefficient [*{of}* *SpeciesName*] =
Scalar_String_Function F = *f*

Scope

Aria Material

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	"string"	–

User_Field

Syntax

Volumetric Heat Transfer Coefficient [*{of}* *SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Volumetric Heat Transfer Coefficient [*{of}* *SpeciesName*] = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Scope

Aria Material

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Volumetric Heat Transfer Coefficient [*{of} SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	“string”	–
plugin_parameters	“string”	–

W

Wall Angle

Correlation

Syntax

Wall Angle [*{of} SpeciesName*] = Correlation ([Theta = *theta*] | [Bulk_Node_Interface = *bulk_node_interface*])

Scope

Aria Material

Summary

Correlation Wall Angle

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
theta	real	0
bulk_node_interface	“string”	–

Wall Length

Correlation

Syntax

Wall Length [*{of}* *SpeciesName*] = Correlation ([L = *l*] | [Bulk_Node_Interface = *bulk_node_interface*])

Scope

Aria Material

Summary

Correlation Wall length

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
l	real	0
bulk_node_interface	“string”	–

Wall Temperature

Constant

Syntax

Wall Temperature [*{of}* *SpeciesName*] = Constant *{tw | value}* = *Tw*

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Tw	real	–

Copied

Syntax

Wall Temperature [*{of}* *SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	“string”	–

Exponential

Syntax

Wall Temperature [*{of}* *SpeciesName*] = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Scope

Aria Material

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Wall Temperature [*{of} SpeciesName*] = Global (Global_Name = *global_name* | [Component = *component*])

Scope

Aria Material

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Wall Temperature [*{of} SpeciesName*] = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Wall Temperature [*{of}* *SpeciesName*] = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Wall Temperature [*{of} SpeciesName*] = Scalar_String_Function F = *f*

Scope

Aria Material

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	"string"	–

User_Field

Syntax

Wall Temperature [*{of}* *SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Wall Temperature [*{of}* *SpeciesName*] = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Scope

Aria Material

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Wall Temperature [*{of} SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	“string”	–
plugin_parameters	“string”	–

Wall Thermal Resistance

Scalar_String_Function

Syntax

Wall Thermal Resistance [*{of} SpeciesName*] = Scalar_String_Function (F = *f* | [Bulk_Node_Interface = *bulk_node_interface*])

Scope

Aria Material

Summary

Scalar string function model for the wall thermal resistance.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	“string”	–
bulk_node_interface	“string”	–

Wetted Perimeter

Constant

Syntax

Wetted Perimeter [*{of}* *SpeciesName*] = Constant *{p | value}* = *p*

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
p	real	–

Copied

Syntax

Wetted Perimeter [*{of}* *SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	“string”	–

Correlation_Spatial_User_Function

Syntax

Wetted Perimeter [*{of}* *SpeciesName*] = Correlation_Spatial_User_Function ([Threshold = *threshold*] | [Multiplier = *multiplier*] | Name = *name* | Bar = *bar* | X = *x* | [Bulk_Node_Interface = *bulk_node_interface*])

Scope

Aria Material

Summary

Correlation spatially varying wetted perimeter based on a coordinate dependent user function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
threshold	real	-1.798e+308
multiplier	real	1
name	“string”	–
bar	“string”	–
x	“string”	–
bulk_node_interface	“string”	–

Exponential

Syntax

Wetted Perimeter [*{of}* *SpeciesName*] = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Scope

Aria Material

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Wetted Perimeter [*{of}* *SpeciesName*] = Global (Global_Name = *global_name* | [Component = *component*])

Scope

Aria Material

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Wetted Perimeter [*{of}* *SpeciesName*] = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Wetted Perimeter [*{of}* *SpeciesName*] = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Wetted Perimeter [*{of}* *SpeciesName*] = Scalar_String_Function F = *f*

Scope

Aria Material

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	"string"	–

Spatial_User_Function

Syntax

Wetted Perimeter [*{of}* *SpeciesName*] = Spatial_User_Function ([Threshold = *threshold*] | [Multiplier = *multiplier*] | Name = *name* | Bar = *bar* | X = *x*)

Scope

Aria Material

Summary

Spatially varying wetted perimeter based on a coordinate dependent user function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
threshold	real	-1.798e+308
multiplier	real	1
name	"string"	–
bar	"string"	–
x	"string"	–

User_Field

Syntax

Wetted Perimeter [*{of}* *SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Wetted Perimeter [*{of}* *SpeciesName*] = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Scope

Aria Material

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1
toggle	"string"	–

User_Plugin

Syntax

Wetted Perimeter [*{of}* *SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	“string”	–
plugin_parameters	“string”	–

Wetting Phase Retardation

Constant

Syntax

Wetting Phase Retardation [*{of} SpeciesName*] = Constant *{r | value}* = *r*

Scope

Aria Material

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
r	real	–

Copied

Syntax

Wetting Phase Retardation [*{of} SpeciesName*] = Copied Source = *source*

Scope

Aria Material

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	“string”	–

Exponential

Syntax

Wetting Phase Retardation [*{of}* *SpeciesName*] = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Scope

Aria Material

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Wetting Phase Retardation [*{of}* *SpeciesName*] = Global (Global_Name = *global_name* | [Component = *component*])

Scope

Aria Material

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Wetting Phase Retardation [*{of}* *SpeciesName*] = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Scope

Aria Material

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Wetting Phase Retardation [*{of}* *SpeciesName*] = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Scope

Aria Material

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Wetting Phase Retardation [*{of} SpeciesName*] = Scalar_String_Function F = *f*

Scope

Aria Material

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	"string"	–

User_Field

Syntax

Wetting Phase Retardation [*{of} SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Scope

Aria Material

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Wetting Phase Retardation [*{of} SpeciesName*] = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Scope

Aria Material

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Wetting Phase Retardation [*{of} SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Scope

Aria Material

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	“string”	–
plugin_parameters	“string”	–

7.7 Rebalance Settings

Combine Rebalance Groups

Syntax

Combine Rebalance Groups *{=} CombinedRebalanceGroups. . .*

Summary

Specifies a list of rebalance groups to combine into the DEFAULT rebalance group for the equation system. By default, each equation system will be split into the following four rebalance groups: DEFAULT, ASSEMBLY, LINEAR_SOLVE, and CHEMEQ_UTIL. Only the ASSEMBLY and CHEMEQ_UTIL rebalance groups can be combined into the DEFAULT rebalance group. The LINEAR_SOLVE rebalance group uses a non-time based measure of the work load and so can not be combined into the DEFAULT

rebalance group. Note that combining rebalance groups may hide load imbalances in the equation system.

Parameter	Value	Default
{=}	{= | are | is}	–
CombinedRebalanceGroups	string. . .	–

Enable Rebalance

Syntax

Enable Rebalance [With Threshold *{=}* *Threshold*]

Summary

Activate rebalance, optionally specifying the rebalance threshold.

Threshold is in terms of the ratio of the maximum (over all processors) cpu time per timestep divided by the average cpu time per timestep. If the observed value of this ratio meets or exceeds the given value, rebalance will be performed.

The default threshold value is 1.25.

Maximum Number Of Nodal Rebalance Iterations

Syntax

Maximum Number Of Nodal Rebalance Iterations *{=}* *Count*

Summary

Specify the maximum number of nodal rebalance iterations to be performed after a dynamic rebalance.

Parameter	Value	Default
{=}	{= | are | is}	–
Count	integer	–

Maximum Number Of Rebalances

Syntax

Maximum Number Of Rebalances *{=}* *Count*

Summary

Specify the maximum number of times to perform a mesh rebalance.

Parameter	Value	Default
{=}	{= | are | is}	–
Count	integer	–

Rebalance After Adaptivity

Syntax

Rebalance After Adaptivity

Summary

Command to automatically execute a rebalance after each solution control adaptivity step

Rebalance Load Measure Type

Syntax

Rebalance Load Measure Type *{=}* *MeasureType*

Summary

Specify the measure used to evaluate the load for rebalancing.

Parameter	Value	Default
{=}	{= | are | is}	–
MeasureType	string	–

Rebalance Time Step Frequency

Syntax

Rebalance Time Step Frequency *{=}* *Frequency*

Summary

Time step frequency for running rebalance.

Parameter	Value	Default
{=}	{= | are | is}	–
Frequency	integer	–

7.8 Solution Control

Solution Control Description

Scope

Procedure

Summary

Contains the commands needed to execute an analysis using the arpeggio procedure that uses Solver Control.

```
begin Solution Control Description Name

   Use System Name

   begin Adaptiveloop Name
   end

   begin Initialize Name
   end

   begin Parameters For
   end

   begin System Name
   end

end Solution Control Description Name
```

Line Commands

Use System

Syntax

Use System *Name*

Summary

This set the name of which system to use.

Parameter	Value	Default
Name	string	–

System

Scope

Solution Control Description

Summary

This block wraps a solver system for a given name. The NAME parameter is the name used to define the system. There can be more than one system block in the Solver Control Description block. The "use system NAME" line command controls which one is to be used.

```
begin System Name

   Adapt Mesh For AdaptRegionName Using AdaptBlockName [When␣
↪WhenExpression]

   Event Name... [ When When-expression  ]

   Output Name [ When When-expression  ]

   Postprocess Aria Region RegionName ([Equation System␣
↪EquationSystemName]  | [When WhenExpression])

   Simulation Max Global Iterations {=} Number

   Simulation Start Time {=} Number

   Simulation Termination Time {=} Number

   Transfer Name [ When When-expression  ]

   Use Initialize Name

   begin Adaptiveloop Name
   end

   begin Sequential Name
   end

   begin Transient Name
   end

end System Name
```

Line Commands

Adapt Mesh

Syntax

Adapt Mesh For *AdaptRegionName* Using *AdaptBlockName* [When *WhenExpression*]

Summary

Adapt the mesh using the adaptive command block name

Parameter	Value	Default
AdaptRegionName	string	–
AdaptBlockName	string	–
WhenExpression	(expression)	–

Event

Syntax

Event *Name. . .* [When *When-expression*]

Summary

Used within a Solver Control block to indicate a single step that has no time associated with it. It can cause a solution transfer between regions or cause something to print.

Parameter	Value	Default
Name	string. . .	–

Output

Syntax

Output *Name* [When *When-expression*]

Summary

A Solver Control Output line command which execute a perform I/O on the region.

Parameter	Value	Default
Name	string	–

Postprocess Aria Region

Syntax

Postprocess Aria Region *RegionName* ([Equation System *EquationSystemName*] | [When *WhenExpression*])

Summary

Run Aria Region postprocessors. If specified, only the single equation system's PPs are run.

Parameter	Value	Default
RegionName	string	–
EquationSystemName	string	–
WhenExpression	(expression)	–

Simulation Max Global Iterations

Syntax

Simulation Max Global Iterations *{=}* *Number*

Summary

The Total number of Solves.

Parameter	Value	Default
{=}	{= | are | is}	–
Number	integer	–

Simulation Start Time

Syntax

Simulation Start Time *{=}* *Number*

Summary

Simulation starting time. (by default 0.0)

Parameter	Value	Default
{=}	{= | are | is}	–
Number	real	–

Simulation Termination Time

Syntax

Simulation Termination Time *{=}* *Number*

Summary

The drop dead time.

Parameter	Value	Default
{=}	{= | are | is}	–
Number	real	–

Transfer

Syntax

Transfer *Name* [When *When-expression*]

Summary

A Solver Control Transfer line command which executes all transfers defined from the specified region. All transfers with a send region of 'name' will be executed.

Parameter	Value	Default
Name	string	–

Use Initialize

Syntax

Use Initialize *Name*

Summary

This set the name of which initialization to use.

Parameter	Value	Default
Name	string	–

Transient

Scope

Adaptiveloop, System

Summary

This block is used to wrap a time loop.

```
begin Transient Name

   Advance Name... [ When When-expression  ]

   Adapt Mesh For AdaptRegionName Using AdaptBlockName [When␣
↪WhenExpression]

   Event Name... [ When When-expression  ]

   Involve Name

   Output Name [ When When-expression  ]

   Postprocess Aria Region RegionName ([Equation System␣
↪EquationSystemName]  | [When WhenExpression])
```

```
Transfer Name [ When When-expression  ]

begin Adaptiveloop Name
end

begin Nonlinear Name
end

begin Subcycle Name
end

end Transient Name
```

Line Commands

Advance

Syntax

Advance *Name...* [When *When-expression*]

Summary

Used within a Solver Control block to indicate a single step that advances the solution. The name is that matches the physics.

Parameter	Value	Default
Name	string...	–

Adapt Mesh

Syntax

Adapt Mesh For *AdaptRegionName* Using *AdaptBlockName* [When *WhenExpression*]

Summary

Adapt the mesh using the adaptive command block name

Parameter	Value	Default
AdaptRegionName	string	–
AdaptBlockName	string	–
WhenExpression	(expression)	–

Event

Syntax

Event *Name. . .* [When *When-expression*]

Summary

Used within a Solver Control block to indicate a single step that has no time associated with it. It can cause a solution transfer between regions or cause something to print.

Parameter	Value	Default
Name	string. . .	–

Involve

Syntax

Involve *Name*

Summary

Specify a physics participant to a coupled problem solved using matrix-free nonlinear.

Parameter	Value	Default
Name	string	–

Output

Syntax

Output *Name* [When *When-expression*]

Summary

A Solver Control Output line command which execute a perform I/O on the region.

Parameter	Value	Default
Name	string	–

Postprocess Aria Region

Syntax

Postprocess Aria Region *RegionName* ([Equation System *EquationSystemName*] | [When *WhenExpression*])

Summary

Run Aria Region postprocessors. If specified, only the single equation system's PPs are run.

Parameter	Value	Default
RegionName	string	–
EquationSystemName	string	–
WhenExpression	(expression)	–

Transfer

Syntax

Transfer *Name* [When *When-expression*]

Summary

A Solver Control Transfer line command which executes all transfers defined from the specified region. All transfers with a send region of 'name' will be executed.

Parameter	Value	Default
Name	string	–

Nonlinear

Scope

Sequential, Transient

Summary

This block is used to wrap a nonlinear solve loop.

```
begin Nonlinear Name

   Advance Name... [ When When-expression  ]

   Adapt Mesh For AdaptRegionName Using AdaptBlockName [When␣
↪WhenExpression]

   Event Name... [ When When-expression  ]

   Involve Name

   Output Name [ When When-expression  ]

   Postprocess Aria Region RegionName ([Equation System␣
↪EquationSystemName]  | [When WhenExpression])

   Transfer Name [ When When-expression  ]
```

```
   begin Subcycle Name
   end

end Nonlinear Name
```

Line Commands

Advance

Syntax

Advance *Name. . .* [When *When-expression*]

Summary

Used within a Solver Control block to indicate a single step that advances the solution. The name is that matches the physics.

Parameter	Value	Default
Name	string. . .	–

Adapt Mesh

Syntax

Adapt Mesh For *AdaptRegionName* Using *AdaptBlockName* [When *WhenExpression*]

Summary

Adapt the mesh using the adaptive command block name

Parameter	Value	Default
AdaptRegionName	string	–
AdaptBlockName	string	–
WhenExpression	(expression)	–

Event

Syntax

Event *Name. . .* [When *When-expression*]

Summary

Used within a Solver Control block to indicate a single step that has no time associated with it. It can cause a solution transfer between regions or cause something to print.

Parameter	Value	Default
Name	string. . .	–

Involve

Syntax

Involve *Name*

Summary

Specify a physics participant to a coupled problem solved using matrix-free nonlinear.

Parameter	Value	Default
Name	string	–

Output

Syntax

Output *Name* [When *When-expression*]

Summary

A Solver Control Output line command which execute a perform I/O on the region.

Parameter	Value	Default
Name	string	–

Postprocess Aria Region

Syntax

Postprocess Aria Region *RegionName* ([Equation System *EquationSystemName*] | [When *WhenExpression*])

Summary

Run Aria Region postprocessors. If specified, only the single equation system's PPs are run.

Parameter	Value	Default
RegionName	string	–
EquationSystemName	string	–
WhenExpression	(expression)	–

Transfer

Syntax

Transfer *Name* [When *When-expression*]

Summary

A Solver Control Transfer line command which executes all transfers defined

from the specified region. All transfers with a send region of 'name' will be executed.

Parameter	Value	Default
Name	string	–

Subcycle

Scope
Nonlinear, Transient

Summary
This block is used to wrap a subcycle time loop.

```
begin Subcycle Name

   Advance Name... [ When When-expression  ]

   Adapt Mesh For AdaptRegionName Using AdaptBlockName [When␣
↪WhenExpression]

   Event Name... [ When When-expression  ]

   Involve Name

   Output Name [ When When-expression  ]

   Postprocess Aria Region RegionName ([Equation System␣
↪EquationSystemName]  | [When WhenExpression])

   Transfer Name [ When When-expression  ]

end Subcycle Name
```

Line Commands

Advance

Syntax
Advance *Name...* [When *When-expression*]

Summary
Used within a Solver Control block to indicate a single step that advances the solution. The name is that matches the physics.

Parameter	Value	Default
Name	string. . .	–

Adapt Mesh

Syntax

Adapt Mesh For *AdaptRegionName* Using *AdaptBlockName* [When *WhenExpression*]

Summary

Adapt the mesh using the adaptive command block name

Parameter	Value	Default
AdaptRegionName	string	–
AdaptBlockName	string	–
WhenExpression	(expression)	–

Event

Syntax

Event *Name. . .* [When *When-expression*]

Summary

Used within a Solver Control block to indicate a single step that has no time associated with it. It can cause a solution transfer between regions or cause something to print.

Parameter	Value	Default
Name	string. . .	–

Involve

Syntax

Involve *Name*

Summary

Specify a physics participant to a coupled problem solved using matrix-free nonlinear.

Parameter	Value	Default
Name	string	–

Output

Syntax

Output *Name* [When *When-expression*]

Summary

A Solver Control Output line command which execute a perform I/O on the region.

Parameter	Value	Default
Name	string	–

Postprocess Aria Region

Syntax

Postprocess Aria Region *RegionName* ([Equation System *EquationSystemName*] | [When *WhenExpression*])

Summary

Run Aria Region postprocessors. If specified, only the single equation system's PPs are run.

Parameter	Value	Default
RegionName	string	–
EquationSystemName	string	–
WhenExpression	(expression)	–

Transfer

Syntax

Transfer *Name* [When *When-expression*]

Summary

A Solver Control Transfer line command which executes all transfers defined from the specified region. All transfers with a send region of 'name' will be executed.

Parameter	Value	Default
Name	string	–

Sequential

Scope

Adaptiveloop, System

Summary

This block is used to wrap a sequential solution. It is used to wrap a sequence of Non-Linear or pseudo time solve step solves.

```
begin Sequential Name

   Advance Name... [ When When-expression  ]

   Adapt Mesh For AdaptRegionName Using AdaptBlockName [When␣
↪WhenExpression]

   Event Name... [ When When-expression  ]

   Involve Name

   Output Name [ When When-expression  ]

   Postprocess Aria Region RegionName ([Equation System␣
↪EquationSystemName]  | [When WhenExpression])

   Transfer Name [ When When-expression  ]

   begin Adaptiveloop Name
   end

   begin Nonlinear Name
   end

end Sequential Name
```

Line Commands

Advance

Syntax

Advance *Name...* [When *When-expression*]

Summary

Used within a Solver Control block to indicate a single step that advances the solution. The name is that matches the physics.

Parameter	Value	Default
Name	string...	–

Adapt Mesh

Syntax

Adapt Mesh For *AdaptRegionName* Using *AdaptBlockName* [When *WhenExpression*]

Summary

Adapt the mesh using the adaptive command block name

Parameter	Value	Default
AdaptRegionName	string	–
AdaptBlockName	string	–
WhenExpression	(expression)	–

Event

Syntax

Event *Name. . .* [When *When-expression*]

Summary

Used within a Solver Control block to indicate a single step that has no time associated with it. It can cause a solution transfer between regions or cause something to print.

Parameter	Value	Default
Name	string. . .	–

Involve

Syntax

Involve *Name*

Summary

Specify a physics participant to a coupled problem solved using matrix-free nonlinear.

Parameter	Value	Default
Name	string	–

Output

Syntax

Output *Name* [When *When-expression*]

Summary

A Solver Control Output line command which execute a perform I/O on the region.

Parameter	Value	Default
Name	string	–

Postprocess Aria Region

Syntax

Postprocess Aria Region *RegionName* ([Equation System *EquationSystemName*] | [When *WhenExpression*])

Summary

Run Aria Region postprocessors. If specified, only the single equation system's PPs are run.

Parameter	Value	Default
RegionName	string	–
EquationSystemName	string	–
WhenExpression	(expression)	–

Transfer

Syntax

Transfer *Name* [When *When-expression*]

Summary

A Solver Control Transfer line command which executes all transfers defined from the specified region. All transfers with a send region of 'name' will be executed.

Parameter	Value	Default
Name	string	–

Initialize

Scope

Solution Control Description

Summary

This block wraps a initializer for a given name. The NAME parameter is the name used to define the initialization block. There can be more than one initialize block in the Solver Control Description block. The "use initialize NAME" line command controls which one is to be used.

```
begin Initialize Name

   Advance Name... [ When When-expression  ]

   Event Name... [ When When-expression  ]

   Involve Name
```

```
   Transfer Name [ When When-expression  ]

end Initialize Name
```

Line Commands

Advance

Syntax

Advance *Name...* [When *When-expression*]

Summary

Used within a Solver Control block to indicate a single step that advances the solution. The name is that matches the physics.

Parameter	Value	Default
Name	string...	–

Event

Syntax

Event *Name...* [When *When-expression*]

Summary

Used within a Solver Control block to indicate a single step that has no time associated with it. It can cause a solution transfer between regions or cause something to print.

Parameter	Value	Default
Name	string...	–

Involve

Syntax

Involve *Name*

Summary

Specify a physics participant to a coupled problem solved using matrix-free nonlinear.

Parameter	Value	Default
Name	string	–

Transfer

Syntax

Transfer *Name* [When *When-expression*]

Summary

A Solver Control Transfer line command which executes all transfers defined from the specified region. All transfers with a send region of 'name' will be executed.

Parameter	Value	Default
Name	string	–

Parameters For

Scope

Solution Control Description

Summary

A Solver Control PARAMETERS block to set up control data for the SC_type parameter. Inside this block one sets the time step parameters or nonlinear parameters.

```
begin Parameters For

   Converged When Convergence-expression

   Incremental Number Of Steps {=} Number

   Initial Deltat {=} Number

   Number Of Adaptivity Steps {=} Number

   Number Of Steps {=} Number

   Reinitialize Transient

   Start Time {=} Number

   Suppress Output From Nonlinear Loop

   Termination Time {=} Number

   Time Step Quantum {=} TimeStepQuantum

   Time Step Style TimeStepStyle...
```

```
Total Change In Time {=} Number

begin Parameters For Aria Region RegionName
end

end Parameters For
```

Line Commands

Converged When

Syntax

Converged When *Convergence-expression*

Summary

Set the convergence expression.

Parameter	Value	Default
Convergence-expression	(expression)	–

Incremental Number Of Steps

Syntax

Incremental Number Of Steps *{=}* *Number*

Summary

The incremental number steps to run the time for nonlinear loop. Number of time steps to run after restarting. NUMBER OF STEPS is total number of steps to run

Parameter	Value	Default
{=}	{= | are | is}	–
Number	integer	–

Initial Deltat

Syntax

Initial Deltat *{=}* *Number*

Summary

Assign an initial delta T

Parameter	Value	Default
{=}	{= | are | is}	–
Number	real	–

Number Of Adaptivity Steps

Syntax

Number Of Adaptivity Steps *{=}* *Number*

Summary

The number steps to run the time or nonlinear loop

Parameter	Value	Default
{=}	{= | are | is}	–
Number	integer	–

Number Of Steps

Syntax

Number Of Steps *{=}* *Number*

Summary

The number steps to run the time for nonlinear loop

Parameter	Value	Default
{=}	{= | are | is}	–
Number	integer	–

Reinitialize Transient

Syntax

Reinitialize Transient

Summary

Reset time and re-initialize regions each step of the adaptivity loop.

Start Time

Syntax

Start Time *{=}* *Number*

Summary

Assign a start time.

Parameter	Value	Default
{=}	{= | are | is}	–
Number	real	–

Suppress Output From Nonlinear Loop

Syntax

Suppress Output From Nonlinear Loop

Summary

Specify that the nonlinear loop will not output. Output will be handled by calls outside of the nonlinear loop (such as an additional advance region call).

Termination Time

Syntax

Termination Time *{=}* *Number*

Summary

Assign a final time to stop

Parameter	Value	Default
{=}	{= | are | is}	–
Number	real	–

Time Step Quantum

Syntax

Time Step Quantum *{=}* *TimeStepQuantum*

Summary

Set the time stepping quantum time for SNAP style stepping.

Parameter	Value	Default
{=}	{= | are | is}	–
TimeStepQuantum	real	–

Time Step Style

Syntax

Time Step Style *TimeStepStyle. . .*

Summary

Set the time stepping style.

When CLIP is specified, the time step size will be clipped at the last step of the transient loop so that it ends at the transient loop's end time. If clip is not specified, the last time is allowed to exceed to the transient loop's end time and the following transient loop will start at the exceeded end time.

When SNAP is specified, the time step is broken down into "quantum" time units. By default this quantum time is 12 orders of magnitude down from the difference between the start and end time for the transient loop. This value can be overridden using the TIME STEP QUANTUM line command. All time values are "snapped" to multiples of the quantum time by rounding to the nearest quantum multiple.

Parameter	Value	Default
TimeStepStyle	{clip | noclip | nosnap | snap}	CLIP NOSNAP

Total Change In Time

Syntax

Total Change In Time *{=}* *Number*

Summary

Use this number and the initial time to compute termination time.

Parameter	Value	Default
{=}	{= | are | is}	–
Number	real	–

Parameters For Aria Region

Scope

Parameters For

Summary

Defines region specific time stepping data

```
begin Parameters For Aria Region RegionName

   Allow Order Switch For HowMany Steps

   Courant Limit {=} Courant_limit

   Fail Time Step On Bad Aztec Solver Status

   Fail Time Step When Time Step Size Ratio Is Below Ratio

   Failed Time Step Size Ratio {=} Ratio

   Initial Nonlinear Residual Tolerance For Time Step Control {=}␣
↪tolerance

   Initial Time Step Size {=} Dt

   Interface Courant Limit {=} Interface Courant_limit

   Known Time Discontinuities {=} times...
```

```
  Limit Solution Increment Field_name {=} Value [ Action  ]

  Maximum Acceptable Linear Residual Ratio {=} Ratio

  Maximum Consecutive Time Step Failures {=} NumFails

  Maximum Global Var Limit Encore_name {=} Value [ Action  ]

  Maximum Linear Residual Ratio {=} Ratio

  Maximum Solution Limit Field_name {=} Value [ Action  ]

  Maximum Time Step Size {=} Dt

  Maximum Time Step Size Ratio {=} Ratio

  Mesh Courant Limit {=} Mesh Courant_limit

  Minimum Global Var Limit Encore_name {=} Value [ Action  ]

  Minimum Resolved Time Step Size {=} Dt

  Minimum Solution Limit Field_name {=} Value [ Action  ]

  Minimum Time Step Size {=} Dt

  Predictor Order {=} order

  Predictor-Corrector Begin After Step {=} Step

  Predictor-Corrector Field Normalization {on} Field_name {=} {courant |␣
↪max | mesh | none | user} [ Scaling  ]

  Predictor-Corrector Normalization {=} {courant | max | mesh | none |␣
↪user} [ Scaling  ]

  Predictor-Corrector Tolerance {=} Predictor_corrector_tolerance

  Reset Initial Time Step Size

  Slope Of Time Step Size {=} slope

  Stop When Initial Nonlinear Residual Is Below Tol

  Time Integration Method {=} {average_acceleration | backward_euler |␣
```

```
↪bdf2 | first_order | midpoint_rule | second_order}

  Time Step Variation {=} {adaptive | fixed}

  Use Initial Nonlinear Residual For Time Step Control {=} {false | no |␣
↪off | on | true | yes}

  Predictor-Corrector Weighting = {nodal_average | volume_weighted}

end Parameters For Aria Region RegionName
```

Line Commands

Allow Order Switch For

Syntax

Allow Order Switch For *HowMany* Steps

Summary

Allow the switching of time integration order from a higher order to first order.

Description

This command allows for the switching of a higher order method to a first order method for a specified number of steps in order to improve the robustness of the simulation especially in the region of discontinuities such as chemistry shutoff. This command is mostly useful in the context of a SECOND_ORDER time integration method.

Parameter	Value	Default
HowMany	integer	–

Courant Limit

Syntax

Courant Limit *{=}* *Courant_limit*

Summary

The Courant Number limit. A value of 0.0 denotes INACTIVE.

Parameter	Value	Default
{=}	{= | are | is}	–
Courant_limit	real	0.0

Fail Time Step On Bad Aztec Solver Status

Syntax

Fail Time Step On Bad Aztec Solver Status

Summary

This command is deprecated and no longer has any effect.

Fail Time Step When Time Step Size Ratio Is Below

Syntax

Fail Time Step When Time Step Size Ratio Is Below *Ratio*

Summary

Specify lower bound for adaptive time step size ratio failure criteria.

If the adaptive time step size ratio falls below this value, Aria will fail the time step. This criteria is specified in terms of the size of the time step size ratio, defined as dt_{n+1}/dt_n. When the adaptive time stepper senses the time step being cut back too much, the time step fails. The default is equal to the FAILED TIME STEP SIZE RATIO.

In simulations where the predicted time step is reduced substantially, a time step may fail repeatedly on this criterion. In that event the user may want to set this Ratio to a small positive value, possibly even zero.

Parameter	Value	Default
Ratio	real	–

Failed Time Step Size Ratio

Syntax

Failed Time Step Size Ratio *{=}* *Ratio*

Summary

Specifies the factor used to cut the time step size when a step fails.

When a time step fails, the time step size is reduced by this factor.

Parameter	Value	Default
{=}	{= | are | is}	–
Ratio	real	0.5

Initial Nonlinear Residual Tolerance For Time Step Control

Syntax

Initial Nonlinear Residual Tolerance For Time Step Control *{=}* *tolerance*

Summary

Set initial nonlinear residual tolerance for time step control.

Description

Sets the tolerance on the initial nonlinear residual to be used for time step control.

Parameter	Value	Default
{=}	{= | are | is}	–
tolerance	real	1.0

Initial Time Step Size

Syntax

Initial Time Step Size *{=}* *Dt*

Summary

Specifies the initial time step size for the given parameter block.

The initial time step size should always be specified for FIXED time stepping and for the starting time block of ADAPTIVE time stepping. When no other command lines affecting timestep are provided in the time step parameter block the time step will remain constant over the time extent of the parameter block.

Parameter	Value	Default
{=}	{= | are | is}	–
Dt	real	–

Interface Courant Limit

Syntax

Interface Courant Limit *{=}* *Interface Courant_limit*

Summary

The Interface Courant Number limit. Applies to both diffuse level set and CDFEM problems. A value of 0.0 denotes INACTIVE.

Parameter	Value	Default
{=}	{= | are | is}	–
Interface Courant_limit	real	0.0

Known Time Discontinuities

Syntax

Known Time Discontinuities *{=}* *times. . .*

Summary

Specifies known time discontinuities in the solution that the adaptive time stepping procedure must capture.

Description

Allows user to specify certain times that the adaptive time stepper must solve at because the solution is known to have a discontinuity there. This is useful for problems where the boundary conditions change suddenly for example.

Parameter	Value	Default
{=}	{= | are | is}	–
times	real. . .	–

Limit Solution Increment

Syntax

Limit Solution Increment *Field_name {=} Value* [*Action*]

Summary

Provides user control for limiting maximum change in a chosen solution Field over a timestep. When action is PREDICT a time step estimation is computed based upon the predicted solution change. When action is FAIL_STEP the time step is still predicted but the the time step will also fail if the suggested solution increment is exceeded.

Description

This command allows one to roughly specify the maximum magnitude that a solution Field can change within a step. A solution increment is estimated based upon the maximum change in the Field in the previous timestep and the proposed adaptive timestep. When the specified value of maximum solution increment is exceeded, the proposed adaptive timestep is reduced in an attempt to limit the solution increment.

One command line should be supplied for each solution Field one wishes to restrict. The restriction is applied directly to the solution Field values, no scaling is invoked.

Parameter	Value	Default
Field_name	string	–
{=}	{= | are | is}	–
Value	real	REAL_MAX

Maximum Acceptable Linear Residual Ratio

Syntax

Maximum Acceptable Linear Residual Ratio *{=} Ratio*

Summary

Specifies a failure criterion for a nonlinear step based on the ratio of the final linear residual to a nonlinear residual.

The ratio must be greater than zero and less than one.

Parameter	Value	Default
{=}	{= | are | is}	–
Ratio	real	–

Maximum Consecutive Time Step Failures

Syntax

Maximum Consecutive Time Step Failures *{=}* *NumFails*

Summary

Specifies a maximum consecutive of time step failures before a simulation is terminated. This prevents the application from continued cutback of the time step when recovery of the solution is unlikely.

Parameter	Value	Default
{=}	{= | are | is}	–
NumFails	integer	20

Maximum Global Var Limit

Syntax

Maximum Global Var Limit *Encore_name* *{=}* *Value* [*Action*]

Summary

This command allows the simulation to continue only when the given Encore variable is lower than a certain magnitude. Inclusion of the optional argument RETRY will cause the solution step to fail and the time step to be reduced. This sequence will be carried out a maximum of three steps, after which the simulation will be terminated.

Description

This command allows the simulation to continue only when the given Encore variable is lower than a certain magnitude.

Parameter	Value	Default
Encore_name	string	–
{=}	{= | are | is}	–
Value	real	–

Maximum Linear Residual Ratio

Syntax

Maximum Linear Residual Ratio *{=}* *Ratio*

Summary

Specifies a failure criterion for a nonlinear step based on the ratio of a nonlinear residual to the initial nonlinear residual.

NOTE: The linear residual is not utilized with this check.

This criteria is not be used unless requested by the user.

Parameter	Value	Default
{=}	{= | are | is}	–
Ratio	real	–

Maximum Solution Limit

Syntax

Maximum Solution Limit *Field_name {=} Value* [*Action*]

Summary

Provides user control for limiting maximum value of a Field solution over a time step. Optional parameters are continue or terminate. When a step fails the step will be cut back and and the optional behavior is invoked when three consecutive fails have occurred. Default behavior is to terminate.

Description

This command allows one to restrict the maximum magnitude that a solution Field can achieve within a simulation by causing the time step to fail when the Field value exceeds a user specified magnitude.

Oftentimes the maximum value will be tied to an event that occurs at a particular magnitude of the Field. This command can be used to impose a different solution behavior when this magnitude of the Field is realized by forcing a failure of the solution step. When the step is failed due to solution limit then cutback of the time step is allowed to occur for two consecutive steps. If the time step is cut back a third time in succession, the specified action of the command will be invoked.

By default the simulation will terminate when the maximum value is reached. If the optional argument is “continue” then time step failure will be over-ridden and the solution sequence will continue, presumably evolving toward lower magnitude values of the Field. This behavior may enable one to overcome large reductions in time step at discontinuous events which may occur for maximum values of the Field.

Parameter	Value	Default
Field_name	string	–
{=}	{= | are | is}	–
Value	real	REAL_MAX

Maximum Time Step Size

Syntax

Maximum Time Step Size *{=}* *Dt*

Summary

Specifies the maximum time step size for this parameter block.

The time step will not be allowed to exceed this value, regardless of adaptive time step selection and regardless of time step requests for other regions (for loosely coupled simulations).

Parameter	Value	Default
{=}	{= | are | is}	–
Dt	real	REAL_MAX

Maximum Time Step Size Ratio

Syntax

Maximum Time Step Size Ratio *{=}* *Ratio*

Summary

Specifies the maximum allowable growth rate of the adaptive time step size.

If the time step selection computes a new time step size that exceeds the previous step size by more than this amount, then the new step size is limited to the product of this factor and the previous step size.

The default is 2.0 for FIRST_ORDER and SECOND_ORDER integration methods.

For BDF2 a value of 2.4 is imposed to enforce stability of the integrator.

Parameter	Value	Default
{=}	{= | are | is}	–
Ratio	real	2.0/2.4

Mesh Courant Limit

Syntax

Mesh Courant Limit *{=}* *Mesh Courant_limit*

Summary

The Mesh Courant Number limit. Applies to problems with mesh motion. CFL condition is applied based on normal mesh velocities at the boundary with the length scale taken as the element volume divided by the element surface area. A value of 0.0 denotes INACTIVE.

Parameter	Value	Default
{=}	{= | are | is}	–
Mesh Courant_limit	real	0.0

Minimum Global Var Limit

Syntax

Minimum Global Var Limit *Encore_name {=} Value* [*Action*]

Summary

This command allows the simulation to continue only when the given Encore variable is higher than a certain magnitude. Inclusion of the optional argument, RETRY, will cause the solution step to fail and the time step to be reduced. This sequence will be carried out a maximum of three steps, after which the simulation will be terminated.

Description

This command allows the simulation to continue only when the given Encore variable is higher than a certain magnitude.

Parameter	Value	Default
Encore_name	string	–
{=}	{= | are | is}	–
Value	real	–

Minimum Resolved Time Step Size

Syntax

Minimum Resolved Time Step Size *{=} Dt*

Summary

Specifies the minimum resolved time step size. If the time step size lies below this value, the step will not fail when the time step size ratio falls too low.

This setting serves as a floor value since the adaptive time stepper will not select a time step size below this value, and any predictor-corrector errors are neglected when the time step size falls below this value.

Parameter	Value	Default
{=}	{= | are | is}	–
Dt	real	0.0

Minimum Solution Limit

Syntax

Minimum Solution Limit *Field_name {=} Value* [*Action*]

Summary

Provides user control for limiting minimum value of a Field solution over a time step. Optional parameters are continue or terminate. When a step fails the step will be cut back and and the optional behavior is invoked when three consecutive fails have occurred. Default behavior is to terminate.

Description

By default the solution minimum limit for all Fields is not active. This command allows one to restrict the minimum magnitude that a solution Field can achieve within a simulation by causing the time step to fail when the Field value goes below a user specified value

See MAXIMUM SOLUTION LIMIT command for more info.

Parameter	Value	Default
Field_name	string	–
{=}	{= | are | is}	–
Value	real	–

Minimum Time Step Size

Syntax

Minimum Time Step Size *{=}* *Dt*

Summary

Specifies the minimum time step size for the given parameter block. The time step will not be allowed to fall below this value, regardless of adaptive time step selection and regardless of time step requests from other regions (for loosely coupled simulations).

Parameter	Value	Default
{=}	{= | are | is}	–
Dt	real	1.0e-16

Predictor Order

Syntax

Predictor Order *{=}* *order*

Summary

Specifies the order of the extrapolation used to predict the solution in transient simulations. An order of 0 corresponds to no extrapolation, instead copying the previous solution into the predicted solution.

Description

This command sets the order of extrapolation in the predictor for transient

problems. A value of 0 basically turns off the extrapolation, and instead copies to old solution to the predicted solution.

The`PREDICTOR ORDER` defaults to 0 for problems if Gas Dynamics equations are defined. Otherwise, it is set to the order of the time integrator for the current time block i.e 1 for first order and 2 for second order

Parameter	Value	Default
{=}	{= | are | is}	–
order	integer	–

Predictor-Corrector Begin After Step

Syntax

Predictor-Corrector Begin After Step *{=}* *Step*

Summary

Specifies after which solution step the Predictor-Corrector timestep calculation will begin to be used in computing an adaptive timestep. Predictor error will always be computed but will not actually be used until the prescribed step is reached.

Default value is 2 for first-order time integration and 3 for second-order time integration. Thus for first-order time integration, a candidate predictor-corrector timestep could appear in step 3.

Parameter	Value	Default
{=}	{= | are | is}	–
Step	integer	2 / 3

Predictor-Corrector Field Normalization

Syntax

Predictor-Corrector Field Normalization *{on}* *Field_name* *{=}* *{courant \| max \| mesh \| none \| user}* [*Scaling*]

Summary

Allows user to change the predictor-corrector normalization type for a given field variable separately from the global option.

Description

Allows user to change the predictor-corrector normalization type for a given field variable separately from the global option.

Normalization types are:

- NONE: Scale by unity
- MAX (default): Scale by the maximum field value

- USER: Scale by a constant (user-supplied) value

Parameter	Value	Default
{on}	{@ | at | for | in | on | over}	–
Field_name	string	–
{=}	{= | are | is}	–
Norm_type	{courant | max | mesh | none | user}	MAX

Predictor-Corrector Normalization

Syntax

Predictor-Corrector Normalization *{=}* *{courant | max | mesh | none | user}* [*Scaling*]

Summary

Provides the ability to specify how the predictor-corrector error is normalized.

Currently supported values are MAX, NONE, and USER. If desired, L2 could be added easily. A possibly better method would be to allow user-specified scales for the variables but that's not yet supported.

The reason for allowing NONE (or even better, a user-specified scale) is to better handle problems that involve something that is zero-based with finite changes. This can happen in ALE problems where the adaptive time stepper is overly restrictive initially when the displacements are all nearly zero, but growing to some finite value.

Default is MAX.

Parameter	Value	Default
{=}	{= | are | is}	–
Normalization	{courant | max | mesh | none | user}	MAX

Predictor-Corrector Tolerance

Syntax

Predictor-Corrector Tolerance *{=}* *Predictor_corrector_tolerance*

Summary

Specifies the tolerance for the difference between the predicted solution and the implicitly solved corrector solution used in adaptive time step selection.

The adaptive time step selection formula is

$$\Delta t_{n+1} = \Delta t_n \left(b \frac{\epsilon}{d_{n+1}} \right)^m$$

where $\Delta t_n \equiv t_{n+1} - t_n$ is the time step size from the most recent solution, $\Delta t_{n+1} \equiv t_{n+2} - t_{n+1}$ is the new time step size, ϵ is the predictor-corrector

tolerance and d_{n+1} is the norm of the difference between the predicted and actual solutions at time t_{n+1}. For first order time integration $m = 1/2$ and $b = 2$. For second order time integration $m = 1/3$ and $b = 3(1 + \Delta t_{n-1}/\Delta t_n)$. See [57].

Parameter	Value	Default
{=}	{= | are | is}	–
Predictor_corrector_tolerance	real	0.001

Reset Initial Time Step Size

Syntax

Reset Initial Time Step Size

Summary

Enforce use of initial time step size specification.

Description

When using adaptive time stepping with restart the initial time step size is taken from the restart file rather than the INITIAL TIME STEP SIZE specification. This choice of time step can be problematic if one wishes to change model features at the beginning of a restart. This command forces the time step selection to respect an initial time step size specification.

Slope Of Time Step Size

Syntax

Slope Of Time Step Size *{=}* *slope*

Summary

Specifies the linear growth rate of the time step size for the given parameter block. The time step size will grow by this amount at each step unless limited by other factors.

Parameter	Value	Default
{=}	{= | are | is}	–
slope	real	–

Stop When Initial Nonlinear Residual Is Below

Syntax

Stop When Initial Nonlinear Residual Is Below *Tol*

Summary

Specify time block termination criteria based upon nonlinear residual value. If the initial nonlinear residual falls below this value, Aria will end the time block. This is useful for obtaining pseudo-transient solutions of steady-state problems.

Parameter	Value	Default
Tol	real	–

Time Integration Method

Syntax

Time Integration Method *{=} {average_acceleration | backward_euler | bdf2 | first_order | midpoint_rule | second_order}*

Summary

Specifies the order of time integration.

Valid options are BACKWARD_EULER, FIRST_ORDER, SECOND_ORDER, BDF2, MIDPOINT_RULE and AVERAGE_ACCELERATION.

Note that the stability criteria of BDF2 will restrict the timestep growth by imposing a MAXIMUM TIME STEP SIZE RATIO of 2.14.

Parameter	Value	Default
{=}	{= | are | is}	–
Time_integration_method	{average_acceleration | backward_euler | bdf2 | first_order | midpoint_rule | second_order}	–

Time Step Variation

Syntax

Time Step Variation *{=} {adaptive | fixed}*

Summary

Specifies how the time step sizes are to be derived. Choose between FIXED and ADAPTIVE time step selection methods.

When the time step is ADAPTIVE the time step is determined using a predictor-corrector error criteria.

Parameter	Value	Default
{=}	{= | are | is}	–
Time_step_variation	{adaptive | fixed}	–

Use Initial Nonlinear Residual For Time Step Control

Syntax

Use Initial Nonlinear Residual For Time Step Control *{=} {false | no | off | on | true | yes}*

Summary

Enable use of initial nonlinear residual as an error estimate to control timestep for loosely coupled problems.

Description

When enabled this option will treat the initial non-linear residual as a proxy for the error in region-region convergence for loosely coupled problems. The time step will be controlled using a predictor-corrector like algorithm.

Parameter	Value	Default
{=}	{= | are | is}	–
enable	{false | no | off | on | true | yes}	–

Predictor-Corrector Weighting

Syntax

Predictor-Corrector Weighting = *{nodal_average | volume_weighted}*

Summary

Use a volume weighted or nodal averaged (default) approach for computing the predictor corrector error

Parameter	Value	Default
WeightingType	{nodal_average | volume_weighted}	–

7.9 Transfer

Transfer

Scope

Procedure

Summary

Transfer region/mesh information. The mechanics/variables information will get sorted out by the calling procedure.

```
begin Transfer Transfer_name

   Abort If Field Not Defined On Copy Transfer Send Or Receive Object

   Abort If Search Object Outside Of Tolerance

   All Fields
```

```
  Copy {surface | volume} {constraints | elements | nodes} From From_
↪region_name To To_region_name

  Distance Function Is Closest Receive Node To Send Centroid

  Exclude Ghosted

  From {constraints | elements | nodes} To {constraints | elements |␣
↪faces | gauss_points | nodes}

  Gauss Point Integration Order {=} Order

  Geometric Tolerance {=} Geometric_tolerance

  Inspect With File {=} File_name Ids {=} ID_list...

  Interpolate {surface | volume} {constraints | elements | nodes} From␣
↪From_region_name To To_region_name

  Interpolation Function Function_Name

  Nodes Outside Region {=} {abort | extrapolate | ignore | project |␣
↪truncate}

  Parametric Tolerance {=} Parametric_tolerance

  Patch Recovery Evaluation {=} {linear least squares | linear moving␣
↪least squares | quadratic least squares | quadratic moving least␣
↪squares}

  Receive Coordinate Transform {=} {add_z0 | identity | remove_z}

  Search Coordinate Field Source_field_name State {new | nm1 | nm2 | nm3␣
↪| nm4 | none | old} To Destination_field_name State {new | nm1 | nm2 |␣
↪nm3 | nm4 | none | old}

  Search Geometric Tolerance {=} Geometric_tolerance

  Search Surface Gap Tolerance {=} Surface_gap_tolerance [ Or Less  ]

  Search Type {=} [ {detailed | parallel | proximity} {detailed |␣
↪parallel | proximity} {detailed | parallel | proximity}  ]

  Send {} Fields
```

```
   Send Block From_blocks... To To_blocks...

   Send Field Source_field_name State {new | nm1 | nm2 | nm3 | nm4 | none␣
↪| old} To Destination_field_name State {new | nm1 | nm2 | nm3 | nm4 |␣
↪none | old} [ Lower Bound Lower_bound Upper Bound Upper_bound  ]

   Toggle Search Warnings {=} {false | no | off | on | true | yes}

   Use Centroid For Geometric Proximity

   begin Receive Blocks
   end

   begin Send Blocks
   end

end Transfer Transfer_name
```

Line Commands

Abort If Field Not Defined On Copy Transfer Send Or Receive Object

Syntax

Abort If Field Not Defined On Copy Transfer Send Or Receive Object

Summary

For testing purposes only. Normally mesh objects in the send or receive mesh which do not have the specified field defined on them are just ignored. This line command allows the construction of tests in which it is known that every mesh object should have the specified field defined on it and to abort if that field is not found.

Abort If Search Object Outside Of Tolerance

Syntax

Abort If Search Object Outside Of Tolerance

Summary

For debugging purposes only. Abort transfer if search object lie outside of specified geometric tolerance.

This command is deprecated. Use “Nodes outside region = abort” instead.

All Fields

Syntax

All Fields

Summary

Select all fields for transfer that have same name and state for source and destination regions.

Copy

Syntax

Copy *{surface | volume} {constraints | elements | nodes}* From *From_region_name* To *To_region_name*

Summary

Copy transfer elements, nodes or constraints from one region to another. The copy transfer is very specific in that the sending and receiving mesh parts must have identical global ids for every element to be copied. The copy transfer works by iterating over all the mesh objects in the receiving mesh and using the global id of the receiving mesh object to find a mesh object in the sending mesh with the same global id. The field to transfer is then copied from the sending to receiving objects. There is no interpolation and the actual coordinates of the sending and receiving objects are not used and could be very different. The copy transfer is used in very special cases where the same mesh was read into both the sending and receiving meshes, there was no element death and there was no adaptivity. In this special case, a copy transfer can be much faster than an interpolation transfer.

Parameter	Value	Default
Option1	{surface | volume}	–
Option2	{constraints | elements | nodes}	–
From_region_name	string	–
To_region_name	string	–

Distance Function Is Closest Receive Node To Send Centroid

Syntax

Distance Function Is Closest Receive Node To Send Centroid

Summary

To be used in conjunction with “SELECT ONE UNIQUE RECEIVER FOR EACH SEND OBJECT”. This helped in the case where the sending and receiving element blocks did not overlap and an element transfer was using element centroids for the distance computation. The elements were very distorted so that a centroid of a surface element could be far from the surface. It was wanted that the receiving element be the one close to the surface of the block and close to the sending element in the adjacent block. Using the corner nodes was enough since it was a tet mesh with plane faces. In this particular and unusual case this alternative method of matching sending and receiving elements was useful, but it is not expected to be used often or maybe never again.

Exclude Ghosted

Syntax

Exclude Ghosted

Summary

exclude ghosted nodes from a copy transfer

From

Syntax

From *{constraints | elements | nodes}* To *{constraints | elements | faces | gauss_points | nodes}*

Summary

Allows the send/receive mesh objects to be different. For a volume element field transfer (e.g. shell) to a face rank field on a surface, use 'from elements to faces'.

Parameter	Value	Default
Option1	{constraints | elements | nodes}	–
Option2	{constraints | elements | faces | gauss_points | nodes}	–

Gauss Point Integration Order

Syntax

Gauss Point Integration Order *{=}* *Order*

Summary

Integration order to use when transferring to Gauss points.

Parameter	Value	Default
{=}	{= | are | is}	–
Order	integer	–

Geometric Tolerance

Syntax

Geometric Tolerance *{=}* *Geometric_tolerance*

Summary

This is the dimensional tolerance applied during the initial (coarse) search. If specified, all the coarse search boxes are padded by this value. The default behavior is for this to be 1e-9 times the problem bounding box size (diagonal) plus a 10 percent relative expansion per element. If a value is specified for this, it is used as a padding in place of the default with no relative box expansion.

During the interpolation transfer there is a geometric search based on the coordinates of the send and receive objects. As part of this search, an axis aligned bounding box is contracted for each sending object and GEOMETRIC

TOLERANCE is used to make this box bigger than just a tight bounding box. Lists of receiving points are then quickly found within these axis aligned boxes.

If all points in the receiving mesh are within at least one box, no additional searching needs to be done and the search algorithm is fast. If there are still points in the receiving mesh that were outside of EVERY box, then a warning message will be issued about an "expensive search for extrapolation" for these points. This 'expensive search" can be very costly if a large number of receiving objects fall into this category and this line command is provided for those special cases.

Parameter	Value	Default
{=}	{= | are | is}	–
Geometric_tolerance	real	–

Inspect With File

Syntax

Inspect With File *{=}* *File_name* Ids *{=}* *ID_list. . .*

Summary

STK Transfer inspection tool that allows user to output the search results for a specified list of entity ids on the receive mesh. The rank of the entities is deduced from the type of transfer being set up.

Parameter	Value	Default
{=}	{= | are | is}	–
File_name	string	–
{=}	{= | are | is}	–
ID_list	integer. . .	–

Interpolate

Syntax

Interpolate *{surface | volume}* *{constraints | elements | nodes}* From *From_region_name* To *To_region_name*

Summary

Interpolate will transfer elements, nodes or constraints from one mesh to another. The interpolation transfer is very general in that the field values to transfer will be interpolated from the sending to receiving mesh based on the coordinates of the sending and receiving mesh objects.

Many line commands can be used to modify the behavior of the interpolation transfer but the basic algorithm is straightforward. Every mesh object in the receiving mesh is converted into a point. For elements this is the average of the nodal coordinates. An element in the sending mesh containing this point is

found. If the field to transfer is nodal, the element shape functions are used to interpolate the nodal field to the receiving point. If the field to transfer is elemental, a bi-linear least squares fit based upon neighboring elements is first performed and then used to define the interpolation of the element field at the receiving point.

Parameter	Value	Default
Option1	{surface | volume}	–
Option2	{constraints | elements | nodes}	–
From_region_name	string	–
To_region_name	string	–

Interpolation Function

Syntax

Interpolation Function *Function_Name*

Summary

Allows an application defined subroutine to be used for the interpolation. Normally the interpolation transfer will determine the best type of interpolation to use for node or element entities: Basis functions for nodal fields and a neighborhood least squares fit for element fields. This command line can be used to override the default behavior if needed. Available interpolation functions are Sum, Copy, Master_Element, Element_Centroid_Constant and Element_Centroid_Linear.

Sum - Sum surrounding entity values on source and send to destination.

Copy - Copy entity values directly from source to destination.

Master_Element - Send interpolated nodal values from nearest source element to destination.

Element_Centroid_Constant - For nodal field transfer send computed centroid value from the nearest source element to destination. For element field transfer send the element field value from the nearest source element to the destination

Element_Centroid_Linear - Send values of interpolated centroid values obtained using patch recovery to destination. Patch recovery involves a least squares reconstruction and evaluation using a patch of elements surrounding the nearest source element. For linear and higher-order reconstructions one should consult the Patch Recovery Evaluation option.

Parameter	Value	Default
Function_Name	string	–

Nodes Outside Region

Syntax

Nodes Outside Region *{=} {abort | extrapolate | ignore | project | truncate}*

Summary

This line command defines what to do when a receiving point is outside the scope of the sending mesh.

IGNORE - The receiving mesh object can be ignored and will receive no value. This is almost never a good idea as it can cause mesh objects just outside to have a zero value when the nodes just inside the mesh might have very large values. This can result in a discontinuous receiving field.

EXTRAPOLATE - This is the default behavior. The sending field is extrapolated beyond the bounds of the sending mesh. This can lead to extrapolation error, such as when a large gradient at the surface causes a negative values when only positive values are acceptable. If this happens to the upper and lower bounds that can be placed on the fields to be transferred with the SEND FIELD command.

TRUNCATE - The receiving coordinate is projected back to the surface of the sending mesh to determine a value. This ensures that the receiving value is not outside of the field values in the sending mesh.

PROJECT - This option is similar to TRUNCATE in which the receiving coordinate is projected back to the surface of the sending mesh to determine a value. In this case more effort is made to make sure that the projection is normal to the surface in the sending mesh. Sometimes gives a better result than Truncate but is a little more expensive to compute.

If the PROJECT option is used in transferring of surface values, the sending mesh should envelope the receiving mesh. Failure to satisfy this condition will generally result in failure of the transfer.

ABORT - If any receiving point is outside the sending mesh by more than the geometric tolerance, abort the simulation. Do not attempt to project, extrapolate, or otherwise handle the point.

Parameter	Value	Default
{=}	{= | are | is}	–
Option	{abort | extrapolate | ignore | project | truncate}	–

Parametric Tolerance

Syntax

Parametric Tolerance *{=} Parametric_tolerance*

Parameter	Value	Default
{=}	{= | are | is}	–
Parametric_tolerance	real	–

Patch Recovery Evaluation

Syntax

Patch Recovery Evaluation *{=} {linear least squares | linear moving least squares | quadratic least squares | quadratic moving least squares}*

Summary

This line command defines the available choices for the patch recovery and evaluation algorithm when using interpolation for element variables. The default option is Linear Least Squares.

Parameter	Value	Default
{=}	{= | are | is}	–
Option	{linear least squares | linear moving least squares | quadratic least squares | quadratic moving least squares}	–

Receive Coordinate Transform

Syntax

Receive Coordinate Transform *{=} {add_z0 | identity | remove_z}*

Summary

Specifies a coordinate transformation to apply to the receiver mesh for transfer. This does not modify the coordinates of the mesh, only the coordinates used for performing the transfer. This option can be used to allow transfers between meshes of different dimensions.

IDENTITY: do not modify the coordinates (default)

REMOVE_Z: For 3D meshes only, removes the z coordinate to enable transferring to
a 2D mesh

ADD_Z0: for 2D meshes only, adds a z coordinate and sets it to zero, enabling transferring
to a 3D mesh.

Parameter	Value	Default
{=}	{= | are | is}	–
Option	{add_z0 | identity | remove_z}	–

Search Coordinate Field

Syntax

Search Coordinate Field *Source_field_name* State *{new | nm1 | nm2 | nm3 | nm4 | none | old}* To *Destination_field_name* State *{new | nm1 | nm2 | nm3 | nm4 | none | old}*

Summary

Normally the interpolation transfers use the default coordinate field to determine geometry information. This line command can be used to specify an alternate field.

Parameter	Value	Default
Source_field_name	string	–
Option1	{new | nm1 | nm2 | nm3 | nm4 | none | old}	–
Destination_field_name	string	–
Option2	{new | nm1 | nm2 | nm3 | nm4 | none | old}	–

Search Geometric Tolerance

Syntax

Search Geometric Tolerance *{=}* *Geometric_tolerance*

Parameter	Value	Default
{=}	{= | are | is}	–
Geometric_tolerance	real	–

Search Surface Gap Tolerance

Syntax

Search Surface Gap Tolerance *{=}* *Surface_gap_tolerance* [Or Less]

Summary

This is the dimensional tolerance applied during the initial (coarse) search. If specified, all the coarse search boxes are padded by this value. The default behavior is for this to be 1e-9 times the problem bounding box size (diagonal) plus a 10 percent relative expansion per element. If a value is specified for this, it is used as a padding in place of the default with no relative box expansion.

This is a tricky parameter best ignored, let it default to something based on the problem size. During the interpolation transfer there is a geometric search based on the coordinates of the send and receive objects. As part of this search, an axis aligned bounding box is contracted for each sending object and SEARCH GAP TOLERANCE is used to make this box bigger than just a tight bounding box. Lists of receiving points are then quickly found within these axis aligned boxes.

If all points in the receiving mesh are within at least one box, no additional searching needs to be done and the search algorithm is fast. If there are still points in the receiving mesh that were outside of EVERY box, then a warning

message will be issued about an “expensive search for extrapolation” for these points. This ‘expensive search” can be very costly if a large number of receiving objects fall into this category and this line command is provided for those special cases.

The OR LESS optional parameter is used when the tolerance must be set to large value for one part of the mesh but much of the mesh needs a much smaller value. In some cases it is necessary for the tolerance to be set to the actual largest surface gap tolerance which may be far too large a gap for the rest of the mesh. Setting OR LESS allows the search tolerance to be reduced in areas of the mesh thus resulting in a faster search.

Parameter	Value	Default
{=}	{= | are | is}	–
Surface_gap_tolerance	real	–

Search Type

Syntax

Search Type *{=}* [*{detailed | parallel | proximity} {detailed | parallel | proximity} {detailed | parallel | proximity}*]

Parameter	Value	Default
{=}	{= | are | is}	–

Send

Syntax

Send *{}* Fields

Summary

Use predefined transfer semantics provided by the specified name.

Parameter	Value	Default
Predefined-transfer	{ }	–

Send Block

Syntax

Send Block *From_blocks. . .* To *To_blocks. . .*

Summary

Add element blocks to a particular same mesh element copy transfer operator.

The copy transfer can have multiple of these lines to define many blocks, but each line sends a single block to a single block:

```
SEND BLOCK block_1 TO block_1
SEND BLOCK block_101 TO block_101
```

The interpolation transfer can have only a single SEND BLOCK line, but can define many from/to blocks:

```
SEND BLOCK block_3 block_5 block_6 TO block_3 block_5
```

Parameter	Value	Default
From_blocks	string. . .	–
To_blocks	string. . .	–

Send Field

Syntax

Send Field *Source_field_name* State *{new | nm1 | nm2 | nm3 | nm4 | none | old}* To *Destination_field_name* State *{new | nm1 | nm2 | nm3 | nm4 | none | old}* [Lower Bound *Lower_bound* Upper Bound *Upper_bound*]

Summary

Specifies the mapping between source and destination field names. Vector and tensor fields can be subscripted using parenthesis and 1's based or brackets and 0 based. Notes on subscripting:

- Does not work for COPY transfers, only INTERPOLATION type transfers.
- If the field name itself actually contains either parenthesis or brackets then we are in trouble and an error is going to be thrown due to a syntax error in index specification.
- Only a single subscript is allowed so vectors of vectors or higher order tensors can not use double subscripts. But it should be possible to determine the correct offset within the field and pick out the correct value with a little effort.
- Once subscripted, only a single value will be transferred. It is not possible to transfer multiple values starting at a certain index, instead multiple line commands must be used, as shown above.
- The indexes can be 0 based with brackets or 1 based when using parenthesis. Although this could be very confusing if mixed within a single line command.
- Both the from and to fields can be subscripted independently on the same line.

example

```
SEND FIELD velocity TO velocity
SEND FIELD temp     TO temperature  lower bound 0
SEND FIELD x        TO y lower bound 10 upper bound 100
SEND FIELD A(2)     TO B(3) lower bound 10 upper bound 100
SEND FIELD A[1]     TO B[2] lower bound 10 upper bound 100
```

Parameter	Value	Default
Source_field_name	string	–
Option1	{new | nm1 | nm2 | nm3 | nm4 | none | old}	–
Destination_field_name	string	–
Option2	{new | nm1 | nm2 | nm3 | nm4 | none | old}	–

Toggle Search Warnings

Syntax

Toggle Search Warnings *{=} {false | no | off | on | true | yes}*

Summary

Specify whether warnings about entities outside of the search domain should be printed. The default behavior is to always print these warning messages.

Parameter	Value	Default
{=}	{= | are | is}	–
Option	{false | no | off | on | true | yes}	–

Use Centroid For Geometric Proximity

Syntax

Use Centroid For Geometric Proximity

Summary

STK Transfer option to trigger the use of centroid based proximity comparison for selecting the best interpolating entity when receive entities lie outside of the domain. Default geometric proximity comparison is based on geometric projection which is more expensive but accurate. The use of this option is a way to reduce computational cost especially for meshes that are fairly regular. However, there is no guarantee of accuracy.

Receive Blocks

Summary

I/O blocks to include in the transfer. Fields defined on these blocks will receive fields from the sending blocks.

```
begin Receive Blocks

   Include All Blocks

   Include Block {=} IncludeBlockList...

   Remove Block {=} ExcludeBlockList...

end Receive Blocks
```

Line Commands

Include All Blocks

Syntax

Include All Blocks

Summary

Use this parameters definition for all blocks. Example:

```
BEGIN SEND BLOCKS
  INCLUDE ALL BLOCKS
  REMOVE BLOCK = block_0 block_1
END SEND BLOCKS
```

Include Block

Syntax

Include Block *{=} IncludeBlockList...*

Summary

List of blocks to exclude. Example:

```
BEGIN SEND BLOCKS
  INCLUDE BLOCK = block_3 block_4
END SEND BLOCKS
```

Parameter	Value	Default
{=}	{= | are | is}	–
IncludeBlockList	string...	–

Remove Block

Syntax

Remove Block *{=}* *ExcludeBlockList...*

Summary

List of blocks to exclude. Example:

```
BEGIN SEND BLOCKS
  INCLUDE ALL BLOCKS
  REMOVE BLOCK = block_0 block_1
END SEND BLOCKS
```

Parameter	Value	Default
{=}	{= | are | is}	–
ExcludeBlockList	string...	–

Send Blocks

Summary

I/O blocks to include in the transfer. Fields defined on these blocks will be transferred to fields in the receiving blocks.

```
begin Send Blocks

   Include All Blocks

   Include Block {=} IncludeBlockList...

   Remove Block {=} ExcludeBlockList...

end Send Blocks
```

Line Commands

Include All Blocks

Syntax

Include All Blocks

Summary

Use this parameters definition for all blocks. Example:

```
BEGIN SEND BLOCKS
  INCLUDE ALL BLOCKS
  REMOVE BLOCK = block_0 block_1
END SEND BLOCKS
```

Include Block

Syntax

Include Block *{=} IncludeBlockList. . .*

Summary

List of blocks to exclude. Example:

```
BEGIN SEND BLOCKS
  INCLUDE BLOCK = block_3 block_4
END SEND BLOCKS
```

Parameter	Value	Default
{=}	{= | are | is}	–
IncludeBlockList	string. . .	–

Remove Block

Syntax

Remove Block *{=} ExcludeBlockList. . .*

Summary

List of blocks to exclude. Example:

```
BEGIN SEND BLOCKS
  INCLUDE ALL BLOCKS
  REMOVE BLOCK = block_0 block_1
END SEND BLOCKS
```

Parameter	Value	Default
{=}	{= | are | is}	–
ExcludeBlockList	string. . .	–

7.10 User Functions

Definition For Function

Scope

Sierra

Summary

Defines a function in terms of its type and values.

```
begin Definition For Function FunctionName

  Abscissa {=} Name...

  Abscissa Offset {=} Abscissa_offset

  Abscissa Scale {=} Abscissa_scale

  At Discontinuity Evaluate To {left | right}

  Column Titles Titles1 Titles2...

  Data File = filename [ X From Column xcol Y From Column ycol  ]

  Debug {=} {off | on}

  Differentiate Expression {=} Expr

  Evaluate Expression {=} Expr

  Evaluate From x0 To x1 By Dx

  Expression Variable: Expr = {element | element_sym_tensor | element_
↪tensor | element_vector | face | global | nodal | nodal_sym_tensor |␣
↪nodal_tensor | nodal_vector} source_var_name... [ State {new | none |␣
↪old}  ]

  Expression Variable: Expr

  Field Types Titles1 Titles2...

  Ordinate {=} Name...

  Ordinate Offset {=} Ordinate_offset
```

```
   Ordinate Scale {=} Ordinate_scale

   Scale By x

   Type {=} {analytic | constant | multicolumn piecewise linear |␣
↪piecewise analytic | piecewise constant | piecewise linear | piecewise␣
↪multivariate | xtable}

   X Offset {=} X_offset

   X Scale {=} X_scale

   Y Offset {=} Y_offset

   Y Scale {=} Y_scale

   begin Expressions empty
   end

   begin Values empty
   end

end Definition For Function FunctionName
```

Line Commands

Abscissa

Syntax

Abscissa *{=}* *Name. . .*

Summary

Specifies a string identifier for the independent variable. Optionally specify a scale and/or offset value which transforms the abscissa values into scaled_abscissa = scale * (abscissa + abscissa_offset).

Parameter	Value	Default
{=}	{= | are | is}	–
Name	string. . .	–

Abscissa Offset

Syntax

Abscissa Offset *{=}* *Abscissa_offset*

Summary

Alias for X OFFSET

Parameter	Value	Default
{=}	{= | are | is}	–
Abscissa_offset	real	–

Abscissa Scale

Syntax

Abscissa Scale *{=} Abscissa_scale*

Summary

Alias for X SCALE

Parameter	Value	Default
{=}	{= | are | is}	–
Abscissa_scale	real	–

At Discontinuity Evaluate To

Syntax

At Discontinuity Evaluate To *{left | right}*

Summary

Control the behavior of a piecewise constant function when evaluated at a discontinuity (plus or minus a small tolerance). The default behavior is to take the value to the right of the discontinuity. If "Left" is specified, the value to the left of the discontinuity is taken instead.

Parameter	Value	Default
Option	{left | right}	–

Column Titles

Syntax

Column Titles *Titles1 Titles2. . .*

Summary

Name the columns (and also defined the expected number of columns) for Multicolumn Piecewise Linear tabular data.

Parameter	Value	Default
Titles	string1 string2. . .	–

Data File

Syntax

Data File = *filename* [X From Column *xcol* Y From Column *ycol*]

Summary

Function will read tabular data from an input file. Compatible with the piecewise linear function type. File must be of form like:

```
-------------------------------------------
\# EXAMPLE FILE
1.099   1191
1.101   221
5.9011  133.1
-------------------------------------------
```

Lines headed by a # are considered comments and will be ignored. Data itself must by in tabular columns separated by whitespace or commas.

Parameter	Value	Default
filename	string	–

Debug

Syntax

Debug *{=} {off | on}*

Summary

Prints functions to the log file.

Parameter	Value	Default
{=}	{= | are | is}	–
Option	{off | on}	–

Differentiate Expression

Syntax

Differentiate Expression *{=} Expr*

Summary

Specifies the expression of derivative of evaluation expression.

Parameter	Value	Default
{=}	{= | are | is}	–
Expr	(expression)	–

Evaluate Expression

Syntax

Evaluate Expression *{=}* *Expr*

Summary

Specifies the expression to evaluate.

Description

This will greatly help with manufactured solutions, and be useful for other purposes as well. This uses the STK expression evaluator to evaluate the provided string. See the STK user manual for details about valid syntax.

```
begin definition for function pressure
  type is analytic
  evaluate expression is "x <= 0.0 ? 0.0 : (x < 0.5 ? x*200.0
     : (x < 1.0 ? (x - 0.5) *50.0 + 100.00 : 150.0))"
end definition for function pressure
```

Parameter	Value	Default
{=}	{= | are | is}	–
Expr	(expression)	–

Evaluate From

Syntax

Evaluate From *x0* To *x1* By *Dx*

Summary

Specifies the range and evaluation interval.

Parameter	Value	Default
x0	real	–
x1	real	–
Dx	real	–

Expression Variable:

Syntax

Expression Variable: *Expr* = *{element | element_sym_tensor | element_tensor | element_vector | face | global | nodal | nodal_sym_tensor | nodal_tensor | nodal_vector}* *source_var_name. . .* [State *{new | none | old}*]

Summary

Specifies what the arguments of an expression correspond to. For example:

```
BEGIN DEFINITION FOR FUNCTION dx_shear
  TYPE = ANALYTIC
```

(continues on next page)

(continued from previous page)

```
  EXPRESSION variable: mx   = NODAL model_coordinates(x)
  EXPRESSION variable: my   = NODAL model_coordinates(y)
  EXPRESSION variable: time = GLOBAL time
  EVALUATE EXPRESSION = "(time/{termTime})*({stretchx}*(mx -␣
↪0.0) + ((my-0.25)/0.5)*{stretchxy})"
END
```

Assuming the above expression is being evaluated on nodes the current values for x and y model coordinates would be placed into mx and my and current analysis time placed into time

Parameter	Value	Default
Expr	string	–
VarType	{element | element_sym_tensor | element_tensor | element_vector | face | global | nodal | nodal_sym_tensor | nodal_tensor | nodal_vector}	–
source_var_string	string...	–

Expression Variable:

Syntax

Expression Variable: *Expr*

Summary

Specifies what the arguments of an expression exists, but does not define it correspond to. For example:

```
BEGIN DEFINITION FOR FUNCTION dx_shear
  TYPE = ANALYTIC
  EXPRESSION variable: mx
  EXPRESSION variable: my
  EXPRESSION variable: time
  EVALUATE EXPRESSION = "(time/{termTime})*({stretchx}*(mx -␣
↪0.0) + ((my-0.25)/0.5)*{stretchxy})"
END
```

Call function must determine what each variable actually is is based off of the string name

Parameter	Value	Default
Expr	string	–

Field Types

Syntax

Field Types *Titles1 Titles2. . .*

Summary

The field types (GLOBAL/NODE/ELEMENT) that correspond to the column titles listed for the multicolumn data.

Parameter	Value	Default
Titles	string1 string2. . .	–

Ordinate

Syntax

Ordinate *{=} Name. . .*

Summary

Specifies a string identifier for the dependent variable. Optionally specify a scale and/or offset value which transforms the ordinate values into scaled_ordinate = scale * (ordinate + ordinate_offset).

Parameter	Value	Default
{=}	{= | are | is}	–
Name	string. . .	–

Ordinate Offset

Syntax

Ordinate Offset *{=} Ordinate_offset*

Summary

Alias for Y OFFSET

Parameter	Value	Default
{=}	{= | are | is}	–
Ordinate_offset	real	–

Ordinate Scale

Syntax

Ordinate Scale *{=} Ordinate_scale*

Summary

Alias for Y SCALE

Parameter	Value	Default
{=}	{= | are | is}	–
Ordinate_scale	real	–

Scale By

Syntax

Scale By *x*

Summary

Specifies a scale factor to be applied.

Parameter	Value	Default
x	real	–

Type

Syntax

Type *{=} {analytic | constant | multicolumn piecewise linear | piecewise analytic | piecewise constant | piecewise linear | piecewise multivariate | xtable}*

Summary

Specifies the type of function.

Parameter	Value	Default
{=}	{= | are | is}	–
Type	{analytic | constant | multicolumn piecewise linear | piecewise analytic | piecewise constant | piecewise linear | piecewise multivariate | xtable}	–

X Offset

Syntax

X Offset *{=} X_offset*

Summary

Sets an offset for the x-axis

Parameter	Value	Default
{=}	{= | are | is}	–
X_offset	real	–

X Scale

Syntax

X Scale *{=} X_scale*

Summary

Sets a scale factor for the x-axis

Parameter	Value	Default
{=}	{= | are | is}	–
X_scale	real	–

Y Offset

Syntax

Y Offset *{=}* *Y_offset*

Summary

Sets an offset for the y-axis

Parameter	Value	Default
{=}	{= | are | is}	–
Y_offset	real	–

Y Scale

Syntax

Y Scale *{=}* *Y_scale*

Summary

Sets a scale factor for the y-axis

Parameter	Value	Default
{=}	{= | are | is}	–
Y_scale	real	–

Values

Scope

Definition For Function, Function

Summary

Lists the values of the function. The values should be listed one pair per line, independent variable first, with whitespace or comma as a separator.

```
begin Values empty

    Xyvalues...

end Values empty
```

Line Commands

Xyvalues

Syntax

Xyvalues *Xyvalues. . .*

Summary

For a piecewise linear function, lists an x-y pair for the nth interpolation point.

Parameter	Value	Default
Xyvalues	real. . .	–

7.11 User Fields

User Field

Syntax

User Field *REAL_or_INTEGER NODE_or_FACE_or_ELEMENT SCALAR_or_VECTOR_or_TENSOR Variable_name* On *Mesh_extent* ([Smooth *nsmooth* Iterations] | [Value *{=}* *Magnitude. . .*])

Summary

Defines a user field of standard type data_type (REAL or INTEGER) on the FEM entity (NODE, FACE, or ELEMENT) with the specified name and Sierra field type (SCALAR, VECTOR, or TENSOR). The field will be defined on the specified mesh part or mesh group alias. Normally this field would be supplied to Aria through a transfer. If a number of smoothing iterations is specified, the field will be smoothed this many times after any transfers.

Parameter	Value	Default
REAL_or_INTEGER	string	–
NODE_or_FACE_or_ELEMENT	string	–
SCALAR_or_VECTOR_or_TENSOR	string	–
Variable_name	string	–
Mesh_extent	string	–
nsmooth	integer	0
{=}	{= | are | is}	–
Magnitude	real. . .	–

7.12 User Variables

User Variable

Scope

Aria Region, Equation System, Explicit Equation System, Root Finder Equation System

Summary

Defines user-named registered variables for this region. The name of the variable is specified by the line parameter 'name'.

Description

This command block allows the user to define their own variables, which have the same status as pre-existing variables. For example, they may be output in the same way as other Aria variables, and may participate in SIERRA restart services. This feature is useful when a user has a need to store data on each mesh object (e.g. nodes) or wishes to evaluate a scalar quantity that requires parallel reduction across processors (e.g. a sum). Typically, a user variable would be set by a SIERRA framework transfer service from another region, or via user subroutines.

```
begin User Variable Name

   Add Part part...

   Get Global From Region other_region RemoteVariable Index

   Global Operator {=} {max | min | sum}

   Initial Value {=} Value...

   Read Variable {=} Variable [ At Time TimeValue  ]

   Reset On {iteration | step}

   Type {=} {edge | element | face | global | node} {integer | real |␣
↪real_tensor | real_vector} [ Length {=} Length  ]

   Use With Restart

end User Variable Name
```

Line Commands

Add Part

Syntax

Add Part *part. . .*

Summary

Adds surface, block or mesh group, by name, to a user variable's extent.

Description

This line command is used to add surfaces or blocks to the extent of a user variable. In Exodus II, surfaces are specified as side sets, that have a global integer identifier. For example, side set 12 would be added by this line command using the surface name surface_12. Similarly element blocks also have a global identifier. Note that in SIERRA, each element of an array of strings must be separated by whitespace.

Parameter	Value	Default
part	string. . .	–

Get Global From Region

Syntax

Get Global From Region *other_region RemoteVariable Index*

Summary

Specifies the user global variable value will be assigned the value of the named RemoteVariable retrieved from another application Region. Index is an offset into the RemoteVariable.

Parameter	Value	Default
other_region	string	–
RemoteVariable	string	–
Index	integer	–

Global Operator

Syntax

Global Operator *{=} {max | min | sum}*

Summary

REQUIRED for GLOBAL variables. Since global variables are often used in user subroutines a need arises in parallel simulations to resolve a representative single GLOBAL variable value across processors. The choice of operator determines how the single global value is to be resolved from each of the individual processor local values of the variable.

For SUM the local global variable values on each processor are summed to obtain a single value of the global variable.

For MIN the local global variable values on each processor are compared and the minimum value over all processors is assigned to the global variable.

For MAX the local global variable values on each processor are compared and the maximum value over all processors is assigned to the global variable.

Parallel synchronization of the single global variables will occur both at the beginning of a nonlinear solution iteration and at the completion of a nonlinear step. Once the synchronization has occurred the local global variable is reset to its initial value.

Parameter	Value	Default
{=}	{= | are | is}	–
UserVarOperator	{max | min | sum}	–

Initial Value

Syntax

Initial Value *{=}* *Value. . .*

Summary

Optional. Specifies the initial value of the user variable. The number of initial values specified with this line command must match the length of the variable specified in the "TYPE" command. If this command is present the specified initial value will automatically be applied to the user variable.

Parameter	Value	Default
{=}	{= | are | is}	–
Value	real. . .	–

Read Variable

Syntax

Read Variable *{=}* *Variable* [At Time *TimeValue*]

Summary

Requests that the variable will be assigned from a time plane of the input ExodusII database. If an AT TIME is not specified, the time plane defaults to the last time plane in the database.

Parameter	Value	Default
{=}	{= | are | is}	–
Variable	string	–

Reset On

Syntax

Reset On *{iteration | step}*

Summary

Specifies when the user global variable will be reset. By default global variable will be reset after the nonlinear step has converged.

Parameter	Value	Default
Assign	{iteration | step}	–

Type

Syntax

Type *{=} {edge | element | face | global | node} {integer | real | real_tensor | real_vector}* [Length *{=} Length*]

Summary

This line command is required. The type of user variable is either real or integer, and is either a single, global value, or is associated with each mesh object of the specified kind. Global variables must have a 'GLOBAL OPERATOR' command associated with them to define its synchronization. The default length is 1. The default initial value for global variables is 0.

Parameter	Value	Default
{=}	{= | are | is}	–
UserVarMeshObject	{edge | element | face | global | node}	–
UserVarType	{integer | real | real_tensor | real_vector}	–

Use With Restart

Syntax

Use With Restart

Summary

Optional. The default behavior is for user-defined variables to not be preserved across a restart. The presence of this line command specifies that the variable is to be written to and restored from the restart file.

7.13 Global Constants

Global Constants

Scope

Sierra

Summary

Set of universal constants for a simulation.

```
begin Global Constants empty

   Faradays Constant {=} Faraday

   Gravity Vector {=} Gravity1 Gravity2 Gravity3

   Ideal Gas Constant {=} Sigma

   K-E Turbulence Model Parameter Param {=} Value

   K-W Turbulence Model Parameter Param {=} Value

   Les Turbulence Model Parameter Param {=} Value

   Light Speed {=} LightSpeed

   Planck Constant {=} PlanckConstant

   Stefan Boltzmann Constant {=} Sigma

   Turbulence Model Param Number {=} Value

end Global Constants empty
```

Line Commands

Faradays Constant

Syntax

Faradays Constant *{=} Faraday*

Summary

Faraday's Constant. **NOTE:** Another Faraday's constant value can be specified while using certain code capabilities. This global constants value will be discarded for any other specified Faraday's constant values.

Parameter	Value	Default
{=}	{= | are | is}	–
Faraday	real	–

Gravity Vector

Syntax

Gravity Vector *{=} Gravity1 Gravity2 Gravity3*

Summary

Gravity constant in vector form, acceleration components.

Parameter	Value	Default
{=}	{= | are | is}	–
Gravity	real1 real2 real3	–

Ideal Gas Constant

Syntax

Ideal Gas Constant *{=} Sigma*

Summary

Ideal gas constant. **NOTE:** Another ideal gas constant value can be specified while using certain code capabilities. This global constants value will be discarded for any other specified ideal gas constant values.

Parameter	Value	Default
{=}	{= | are | is}	–
Sigma	real	–

K-E Turbulence Model Parameter

Syntax

K-E Turbulence Model Parameter *Param {=} Value*

Summary

$k-\epsilon$ RANS turbulence model parameters.

Parameter	Value	Default
Param	string	–
{=}	{= | are | is}	–
Value	real	–

K-W Turbulence Model Parameter

Syntax

K-W Turbulence Model Parameter *Param* *{=}* *Value*

Summary

$k-\omega$ RANS turbulence model parameters.

Parameter	Value	Default
Param	string	–
{=}	{= | are | is}	–
Value	real	–

Les Turbulence Model Parameter

Syntax

Les Turbulence Model Parameter *Param* *{=}* *Value*

Summary

LES turbulence model parameters.

Parameter	Value	Default
Param	string	–
{=}	{= | are | is}	–
Value	real	–

Light Speed

Syntax

Light Speed *{=}* *LightSpeed*

Summary

Speed of Light. Depending on the units involved in the specific problem by the user, this value will differ.

Parameter	Value	Default
{=}	{= | are | is}	–
LightSpeed	real	–

Planck Constant

Syntax

Planck Constant *{=}* *PlanckConstant*

Summary

Planck Constant. Depending on the units involved in the specific problem by the user, this value will differ.

Parameter	Value	Default
{=}	{= | are | is}	–
PlanckConstant	real	–

Stefan Boltzmann Constant

Syntax

Stefan Boltzmann Constant *{=}* *Sigma*

Summary

Stefan-Boltzmann constant. Depending on the units involved in the specific problem by the user, this value will differ.

Parameter	Value	Default
{=}	{= | are | is}	–
Sigma	real	–

Turbulence Model

Syntax

Turbulence Model *Param* Number *{=}* *Value*

Summary

Turbulence model Schmidt and Prandtl numbers

Parameter	Value	Default
Param	string	–
{=}	{= | are | is}	–
Value	real	–

Aria Equation Setup Commands

7.14 Boundary Conditions

Generic BC Models

A list of available boundary condition types is provided below

Bc Bar_Flux

Energy

Generalized_Nat_Conv

Syntax

Bc Bar_Flux For Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Generalized_Nat_Conv (Bar = *bar* | [Hcorr = *hcorr*] | [H = *h*] | [Bar_Block_Name = *bar_block_name*] | [Command_Block_Name = *command_block_name*] | [Power_Output = *power_output*] | [Flux_Output = *flux_output*] | [Multiplier = *multiplier*])

Summary

This boundary condition couples the ENERGY equation to the 1D advective bar energy equation.

Description

The flux applied to the advective bar equation is specified as

$$-\mathbf{q} \cdot \hat{\mathbf{n}} = -h\,(T_{bar} - T)$$

where h is the heat transfer coefficient, T_{bar} is the bar temperature and T is the temperature of the surface to which the flux is applied. The heat transfer coefficient is specified through the optional parameters “hcorr” or “h”, where the former must name a heat transfer correlation coefficient block in the input deck and where the latter is a positive constant. Note the flux applied to the bar is equal and opposite to the flux applied to the surface.

The required parameter “bar” must refer to an advective bar specified in the input deck.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output

If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Flux_Output

If specified, the average flux of the given BC/Src is calculated and stored into the global variable name specified as the *flux_output* argument.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
bar	"string"	–
hcorr	"string"	–
h	"string"	–
bar_block_name	"string"	–
command_block_name	"string"	–
power_output	"string"	–
flux_output	"string"	–
multiplier	real	1

Enthalpy

Generalized_Nat_Conv

Syntax

Bc Bar_Flux For Enthalpy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Generalized_Nat_Conv (Bar = *bar* | [Hcorr = *hcorr*] | [H = *h*] | [Bar_Block_Name = *bar_block_name*] | [Command_Block_Name = *command_block_name*] | [Power_Output = *power_output*] | [Flux_Output = *flux_output*] | [Multiplier = *multiplier*])

Summary

This boundary condition couples the ENTHALPY equation to the 1D advective bar energy equation.

Description

The flux applied to the advective bar equation is specified as

$$-\mathbf{q} \cdot \hat{\mathbf{n}} = -h\,(T_{bar} - T)$$

where h is the heat transfer coefficient, T_{bar} is the bar temperature and T is the temperature of the surface to which the flux is applied. The heat transfer coefficient is specified through the optional parameters "hcorr" or "h", where

the former must name a heat transfer correlation coefficient block in the input deck and where the latter is a positive constant. Note the flux applied to the bar is equal and opposite to the flux applied to the surface.

The required parameter "bar" must refer to an advective bar specified in the input deck.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output

If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Flux_Output

If specified, the average flux of the given BC/Src is calculated and stored into the global variable name specified as the *flux_output* argument.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
bar	"string"	–
hcorr	"string"	–
h	"string"	–
bar_block_name	"string"	–
command_block_name	"string"	–
power_output	"string"	–
flux_output	"string"	–
multiplier	real	1

Porous_Enthalpy

Generalized_Nat_Conv

Syntax

Bc Bar_Flux For Porous_Enthalpy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Generalized_Nat_Conv (Bar = *bar* | [Hcorr = *hcorr*] | [H = *h*] | [Bar_Block_Name = *bar_block_name*] | [Command_Block_Name = *command_block_name*] | [Power_Output = *power_output*] | [Flux_Output = *flux_output*] | [Multiplier = *multiplier*])

Summary

This boundary condition couples the 1D advective bar energy equation to the porous enthalpy equation.

Description

For a single temperature porous media problem, the flux applied to the advective bar equation is specified as

$$-\mathbf{q} \cdot \hat{\mathbf{n}} = -h\,(T_{bar} - T)$$

where h is the heat transfer coefficient, T_{bar} is the bar temperature and T is the temperature of the porous media surface. The heat transfer coefficient is specified through the optional parameters “hcorr” or “h”, where the former must name a heat transfer correlation coefficient block in the input deck and where the latter is a positive constant.

For multi-phase porous flows, the heat flux applied to the advective bar equation will be a weighted sum of the flux applied to each individual phase. For example, applying this flux to both phases in a two-phase porous media (solid and liquid) will result in the following bar flux

$$-\mathbf{q} \cdot \hat{\mathbf{n}} = -h\,(T_{bar} - T_l)\,f_l - h\,(T_{bar} - T_s)\,f_s$$

where the first term and second term are equal and opposite to the heat flux received by the liquid and solid phases, respectively. Here, f_l and f_s correspond to the bar flux contribution factors specified in the material block of each phase. If the bar flux contribution factor is not specified, a default weight of one will be used.

The required parameter “bar” must refer to an advective bar specified in the input deck.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output

If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Flux_Output

If specified, the average flux of the given BC/Src is calculated and stored into the global variable name specified as the *flux_output* argument.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
bar	"string"	–
hcorr	"string"	–
h	"string"	–
bar_block_name	"string"	–
command_block_name	"string"	–
power_output	"string"	–
flux_output	"string"	–
multiplier	real	1

Bc Bar_No_Coverage_Flux

Energy

Generalized_Nat_Conv

Syntax

Bc Bar_No_Coverage_Flux For Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Generalized_Nat_Conv (Bar = *bar* | [Hcorr = *hcorr*] | [H = *h*] | [Bar_Block_Name = *bar_block_name*] | [Command_Block_Name = *command_block_name*] | [Power_Output = *power_output*] | [Flux_Output = *flux_output*] | [Multiplier = *multiplier*])

Summary

This boundary condition couples the ENERGY equation to the 1D advective bar energy equation.

Description

The flux applied to the advective bar equation is specified as

$$-\mathbf{q} \cdot \hat{\mathbf{n}} = -h\,(T_{bar} - T)$$

where h is the heat transfer coefficient, T_{bar} is the bar temperature and T is the temperature of the surface to which the flux is applied. The heat transfer coefficient is specified through the optional parameters "hcorr" or "h", where the former must name a heat transfer correlation coefficient block in the input deck and where the latter is a positive constant. Note the flux applied to the bar is equal and opposite to the flux applied to the surface.

The required parameter "bar" must refer to an advective bar specified in the input deck.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output

If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Flux_Output

If specified, the average flux of the given BC/Src is calculated and stored into the global variable name specified as the *flux_output* argument.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
bar	"string"	–
hcorr	"string"	–
h	"string"	–
bar_block_name	"string"	–
command_block_name	"string"	–
power_output	"string"	–
flux_output	"string"	–
multiplier	real	1

Enthalpy

Generalized_Nat_Conv

Syntax

Bc Bar_No_Coverage_Flux For Enthalpy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Generalized_Nat_Conv (Bar = *bar* | [Hcorr = *hcorr*] | [H = *h*] | [Bar_Block_Name = *bar_block_name*] | [Command_Block_Name = *command_block_name*] | [Power_Output = *power_output*] | [Flux_Output = *flux_output*] | [Multiplier = *multiplier*])

Summary

This boundary condition couples the ENTHALPY equation to the 1D advective bar energy equation.

Description

The flux applied to the advective bar equation is specified as

$$-\mathbf{q} \cdot \hat{\mathbf{n}} = -h\,(T_{bar} - T)$$

where h is the heat transfer coefficient, T_{bar} is the bar temperature and T is the temperature of the surface to which the flux is applied. The heat transfer coefficient is specified through the optional parameters "hcorr" or "h", where

the former must name a heat transfer correlation coefficient block in the input deck and where the latter is a positive constant. Note the flux applied to the bar is equal and opposite to the flux applied to the surface.

The required parameter "bar" must refer to an advective bar specified in the input deck.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output
If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Flux_Output
If specified, the average flux of the given BC/Src is calculated and stored into the global variable name specified as the *flux_output* argument.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
bar	"string"	–
hcorr	"string"	–
h	"string"	–
bar_block_name	"string"	–
command_block_name	"string"	–
power_output	"string"	–
flux_output	"string"	–
multiplier	real	1

Porous_Enthalpy

Generalized_Nat_Conv

Syntax

Bc Bar_No_Coverage_Flux For Porous_Enthalpy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Generalized_Nat_Conv (Bar = *bar* | [Hcorr = *hcorr*] | [H = *h*] | [Bar_Block_Name = *bar_block_name*] | [Command_Block_Name = *command_block_name*] | [Power_Output = *power_output*] | [Flux_Output = *flux_output*] | [Multiplier = *multiplier*])

Summary

This boundary condition couples the 1D advective bar energy equation to the porous enthalpy equation.

Description

For a single temperature porous media problem, the flux applied to the advective bar equation is specified as

$$-\mathbf{q} \cdot \hat{\mathbf{n}} = -h\,(T_{bar} - T)$$

where h is the heat transfer coefficient, T_{bar} is the bar temperature and T is the temperature of the porous media surface. The heat transfer coefficient is specified through the optional parameters “hcorr” or “h”, where the former must name a heat transfer correlation coefficient block in the input deck and where the latter is a positive constant.

For multi-phase porous flows, the heat flux applied to the advective bar equation will be a weighted sum of the flux applied to each individual phase. For example, applying this flux to both phases in a two-phase porous media (solid and liquid) will result in the following bar flux

$$-\mathbf{q} \cdot \hat{\mathbf{n}} = -h\,(T_{bar} - T_l)\,f_l - h\,(T_{bar} - T_s)\,f_s$$

where the first term and second term are equal and opposite to the heat flux received by the liquid and solid phases, respectively. Here, f_l and f_s correspond to the bar flux contribution factors specified in the material block of each phase. If the bar flux contribution factor is not specified, a default weight of one will be used.

The required parameter “bar” must refer to an advective bar specified in the input deck.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output

If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Flux_Output

If specified, the average flux of the given BC/Src is calculated and stored into the global variable name specified as the *flux_output* argument.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
bar	"string"	–
hcorr	"string"	–
h	"string"	–
bar_block_name	"string"	–
command_block_name	"string"	–
power_output	"string"	–
flux_output	"string"	–
multiplier	real	1

Bc Bulk_Node_Coupling

Density

K_Factor_Flow

Syntax

Bc Bulk_Node_Coupling For Density ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = K_Factor_Flow [Flow_Rate_Limiter = *flow_rate_limiter*]

Summary

Volumetric flow rate, Q, calculated as $Q = K\sqrt{\frac{P_{high}-P_{low}}{\rho}}$. Absent any other sources, the maximum flow rate is defined based on the mass loss to achieve

pressure equilibrium and the fluid time step. If the optional flow rate limiter is specified, the flow rate is limited by the product of the limiter factor and the maximum flow rate without any other sources.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
flow_rate_limiter	real	0

K_Factor_Flow_With_Choking

Syntax

Bc Bulk_Node_Coupling For Density ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = K_Factor_Flow_With_Choking (Critical_Pressure_Ratio = *critical_pressure_ratio* | [Flow_Rate_Limiter = *flow_rate_limiter*])

Summary

Volumetric flow rate, Q, calculated as $Q = K\sqrt{\frac{(P_{high}-P_{low})}{\rho}}$ for $\frac{P_{high}}{P_{low}} < R_p$, where R_p is the critical_pressure_ratio. Above R_p the flow rate is calculated as $Q = K * \sqrt{\frac{P_{high}(1-1/R_p)}{\rho}}$. Absent any other sources, the maximum flow rate is defined based on the mass loss to achieve pressure equilibrium and the fluid time step. If the optional flow rate limiter is specified, the flow rate is limited by the product of the limiter factor and the maximum flow rate without any other sources.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
critical_pressure_ratio	real	–
flow_rate_limiter	real	0

Species

K_Factor_Flow

Syntax

Bc Bulk_Node_Coupling For Species ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = K_Factor_Flow [Flow_Rate_Limiter = *flow_rate_limiter*]

Summary

Volumetric flow rate, Q, calculated as $Q = K\sqrt{\frac{P_{high}-P_{low}}{\rho}}$. Absent any other sources, the maximum flow rate is defined based on the mass loss to achieve pressure equilibrium and the fluid time step. If the optional flow rate limiter is specified, the flow rate is limited by the product of the limiter factor and the maximum flow rate without any other sources.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
flow_rate_limiter	real	0

K_Factor_Flow_With_Choking

Syntax

Bc Bulk_Node_Coupling For Species ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = K_Factor_Flow_With_Choking (Critical_Pressure_Ratio = *critical_pressure_ratio* | [Flow_Rate_Limiter = *flow_rate_limiter*])

Summary

Volumetric flow rate, Q, calculated as $Q = K\sqrt{\frac{(P_{high}-P_{low})}{\rho}}$ for $\frac{P_{high}}{P_{low}} < R_p$, where R_p is the critical_pressure_ratio. Above R_p the flow rate is calculated as $Q = K * \sqrt{\frac{P_{high}(1-1/R_p)}{\rho}}$. Absent any other sources, the maximum flow rate is defined based on the mass loss to achieve pressure equilibrium and the fluid time step. If the optional flow rate limiter is specified, the flow rate is limited by the product of the limiter factor and the maximum flow rate without any other sources.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
critical_pressure_ratio	real	–
flow_rate_limiter	real	0

Bc Bulknode_Flux

Cvfem_Energy

Conductance

Syntax

Bc Bulknode_Flux For Cvfem_Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Conductance ([Bulk_Node = *bulk_node*] | [Toggle = *toggle*])

Summary

Applies a contact heat flux of the form $C(T_1 - T_2)$ to sides 1 and 2 of the contact interface where C is a user specified conductance coefficient. Used in conjunction with a CONDUCTANCE COEFFICIENT model.

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
bulk_node	"string"	–
toggle	"string"	–

Scalar_String_Function

Syntax

Bc Bulknode_Flux For Cvfem_Energy ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function (Bulk_Node = *bulk_node* | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
bulk_node	"string"	–
f	"string"	–

Energy

Generalized_Nat_Conv

Syntax

Bc Bulknode_Flux For Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Generalized_Nat_Conv (Bulk_Node = *bulk_node* | [H = *h*] | [Hcorr = *hcorr*] | [Toggle = *toggle*])

Summary

Applies a heat flux of the form $J \cdot n = h(T - T_{bulk})$ where h is the convection coefficient defined as an inline string function, a property of the surface material, or the heat transfer correlation specified by the optional 'hcorr' parameter.

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
bulk_node	“string”	–
h	“string”	–
hcorr	“string”	–
toggle	“string”	–

Generalized_Rad

Syntax

Bc Bulknode_Flux For Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Generalized_Rad (Bulk_Node = *bulk_node* | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Summary

Applies a heat flux of the form $J \cdot n = \sigma \epsilon F (T^4 - T_{bulk}^4)$ where ϵ is the emissivity defined in the solid material model and F is the radiation form factor defined in the surface material (optional, defaults to 1).

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
bulk_node	"string"	–
multiplier	real	1
toggle	"string"	–

Scalar_String_Function

Syntax

Bc Bulknode_Flux For Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function (Bulk_Node = *bulk_node* | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
bulk_node	"string"	–
f	"string"	–

Mass_Balance

Bulk_Node_Mass_Open

Syntax

Bc Bulknode_Flux For Mass_Balance ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Bulk_Node_Mass_Open Bulk_Node = *bulk_node*

Summary

Open BC between a porous medium and a bulk node

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
bulk_node	“string”	–

Bulk_Node_Open

Syntax

Bc Bulknode_Flux For Mass_Balance ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Bulk_Node_Open Bulk_Node = *bulk_node*

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
bulk_node	“string”	–

Bulk_Node_Species_Open

Syntax

Bc Bulknode_Flux For Mass_Balance ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Bulk_Node_Species_Open Bulk_Node = *bulk_node*

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
bulk_node	“string”	–

K_Factor

Syntax

Bc Bulknode_Flux For Mass_Balance ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = K_Factor (Bulk_Node = *bulk_node* | [Flow_Rate_Limiter = *flow_rate_limiter*])

Summary

Volumetric flow rate, Q, calculated as $Q = K\sqrt{\frac{P_{high}-P_{low}}{\rho}}$. Absent any other sources, the maximum flow rate is defined based on the mass loss to achieve pressure equilibrium and the fluid time step. If the optional flow rate limiter is specified, the flow rate is limited by the product of the limiter factor and the maximum flow rate without any other sources.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
bulk_node	"string"	–
flow_rate_limiter	real	0

K_Factor_With_Choking

Syntax

Bc Bulknode_Flux For Mass_Balance ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = K_Factor_With_Choking (Bulk_Node = *bulk_node* | Critical_Pressure_Ratio = *critical_pressure_ratio* | [Flow_Rate_Limiter = *flow_rate_limiter*])

Summary

Volumetric flow rate, Q, calculated as $Q = K\sqrt{\frac{(P_{high}-P_{low})}{\rho}}$ for $\frac{P_{high}}{P_{low}} < R_p$, where R_p is the critical_pressure_ratio. Above R_p the flow rate is calculated as $Q = K * \sqrt{\frac{P_{high}(1-1/R_p)}{\rho}}$. Absent any other sources, the maximum flow rate is defined based on the mass loss to achieve pressure equilibrium and the fluid time step. If the optional flow rate limiter is specified, the flow rate is limited by the product of the limiter factor and the maximum flow rate without any other sources.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
bulk_node	"string"	–
critical_pressure_ratio	real	–
flow_rate_limiter	real	0

Scalar_String_Function

Syntax

Bc Bulknode_Flux For Mass_Balance ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function (Bulk_Node = *bulk_node* | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
bulk_node	"string"	–
f	"string"	–

Porous_Enthalpy

Bulk_Node_Open

Syntax

Bc Bulknode_Flux For Porous_Enthalpy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Bulk_Node_Open Bulk_Node = *bulk_node*

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
bulk_node	“string”	–

Contact_Resistance

Syntax

Bc Bulknode_Flux For Porous_Enthalpy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Contact_Resistance ([Opposite_Material_Phase = *opposite_material_phase*] | [Bulk_Node = *bulk_node*])

Summary

Applies a heat flux of the form $J \cdot n = h(T - T_{opp})$ where h is the gap conductance coefficient defined for the surface material and the material phase of T_{opp} is optionally specified.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
opposite_material_phase	"string"	–
bulk_node	"string"	–

Generalized_Nat_Conv

Syntax

Bc Bulknode_Flux For Porous_Enthalpy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Generalized_Nat_Conv (Bulk_Node = *bulk_node* | [H = *h*] | [Hcorr = *hcorr*] | [Toggle = *toggle*])

Summary

Applies a heat flux of the form $J \cdot n = h(T - T_{bulk})$ where h is the convection coefficient defined as an inline string function, a property of the surface material, or the heat transfer correlation specified by the optional 'hcorr' parameter.

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
bulk_node	"string"	–
h	"string"	–
hcorr	"string"	–
toggle	"string"	–

Scalar_String_Function

Syntax

Bc Bulknode_Flux For Porous_Enthalpy ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function (Bulk_Node = *bulk_node* | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
bulk_node	"string"	–
f	"string"	–

Porous_Species

Bulk_Node_Open_All_Inflow

Syntax

Bc Bulknode_Flux For Porous_Species ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Bulk_Node_Open_All_Inflow Bulk_Node = *bulk_node*

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
bulk_node	“string”	–

Scalar_String_Function

Syntax

Bc Bulknode_Flux For Porous_Species ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function (Bulk_Node = *bulk_node* | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
bulk_node	“string”	–
f	“string”	–

Bc Colloc_Rotated

Cvfem_Mesh

Constant

Syntax

Bc Colloc_Rotated For Cvfem_Mesh ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Unrotated = *unrotated*] | [Valuex = *valuex*] | [Valuey = *valuey*] | [Valuez = *valuez*])

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
unrotated	“string”	–
valuex	real	0
valuey	real	0
valuez	real	0

Kinematic

Syntax

Bc Colloc_Rotated For Cvfem_Mesh ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Kinematic ([Unrotated = *unrotated*] | [V0 = *v0*] | [Shared_Normal = *shared_normal*] | [No_Mesh_Movement = *no_mesh_movement*])

Summary

Kinematic model for rotated CVFEM_MESH boundary condition

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
unrotated	“string”	–
v0	real	0
shared_normal	“string”	–
no_mesh_movement	“string”	–

Pyrolysis

Syntax

Bc Colloc_Rotated For Cvfem_Mesh ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Pyrolysis ([Unrotated = *unrotated*] | [T_Ref = *t_ref*] | K = *k* | K_0 = *k_0* | N = *n* | [Temperature = *temperature*])

Summary

Pyrolysis model for rotated CVFEM_MESH boundary condition

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
unrotated	“string”	–
t_ref	real	0
k	real	–
k_0	real	–
n	real	–
temperature	“string”	–

Recession

Syntax

Bc Colloc_Rotated For Cvfem_Mesh ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Recession ([Unrotated = *unrotated*] | [P_Ref = *p_ref*] | K = *k* | K_0 = *k_0* | N = *n* | [Pressure = *pressure*])

Summary

Recession model for rotated CVFEM_MESH boundary condition

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
unrotated	"string"	–
p_ref	real	0
k	real	–
k_0	real	–
n	real	–
pressure	"string"	–

Vector_String_Function

Syntax

Bc Colloc_Rotated For Cvfem_Mesh ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Vector_String_Function ([Unrotated = *unrotated*] | F_X = *f_x* | F_Y = *f_y* | [F_Z = *f_z*])

Summary

Value from a string function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
unrotated	"string"	–
f_x	"string"	–
f_y	"string"	–
f_z	"string"	–

Cvfem_Mesh_X

Constant

Syntax

Bc Colloc_Rotated For Cvfem_Mesh_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Unrotated = *unrotated*] | Value = *value*)

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
unrotated	“string”	–
value	real	–

Exponential

Syntax

Bc Colloc_Rotated For Cvfem_Mesh_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Unrotated = *unrotated*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
unrotated	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Colloc_Rotated For Cvfem_Mesh_X ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Unrotated = *unrotated*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
unrotated	"string"	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Bc Colloc_Rotated For Cvfem_Mesh_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Unrotated = *unrotated*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
unrotated	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc Colloc_Rotated For Cvfem_Mesh_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Unrotated = *unrotated*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
unrotated	"string"	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Colloc_Rotated For Cvfem_Mesh_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Unrotated = *unrotated*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name.

Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
unrotated	"string"	–
f	"string"	–

User_Field

Syntax

Bc Colloc_Rotated For Cvfem_Mesh_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Unrotated = *unrotated*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
unrotated	"string"	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc Colloc_Rotated For Cvfem_Mesh_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Unrotated = *unrotated*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
unrotated	“string”	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Bc Colloc_Rotated For Cvfem_Mesh_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Unrotated = *unrotated*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
unrotated	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

Cvfem_Mesh_Y

Constant

Syntax

Bc Colloc_Rotated For Cvfem_Mesh_Y ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Unrotated = *unrotated*] | Value = *value*)

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
unrotated	"string"	–
value	real	–

Exponential

Syntax

Bc Colloc_Rotated For Cvfem_Mesh_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Unrotated = *unrotated*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
unrotated	“string”	–
variable	“string”	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Colloc_Rotated For Cvfem_Mesh_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Unrotated = *unrotated*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
unrotated	"string"	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Bc Colloc_Rotated For Cvfem_Mesh_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Unrotated = *unrotated*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
unrotated	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc Colloc_Rotated For Cvfem_Mesh_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Unrotated = *unrotated*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
unrotated	"string"	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Colloc_Rotated For Cvfem_Mesh_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Unrotated = *unrotated*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
unrotated	“string”	–
f	“string”	–

User_Field

Syntax

Bc Colloc_Rotated For Cvfem_Mesh_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Unrotated = *unrotated*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
unrotated	“string”	–
name	“string”	–
scaling	real	1
global_var	“string”	–
time_function	“string”	–

User_Function

Syntax

Bc Colloc_Rotated For Cvfem_Mesh_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Unrotated = *unrotated*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
unrotated	“string”	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Bc Colloc_Rotated For Cvfem_Mesh_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Unrotated = *unrotated*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
unrotated	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

Cvfem_Mesh_Z

Constant

Syntax

Bc Colloc_Rotated For Cvfem_Mesh_Z ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Unrotated = *unrotated*] | Value = *value*)

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
unrotated	"string"	–
value	real	–

Exponential

Syntax

Bc Colloc_Rotated For Cvfem_Mesh_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Unrotated = *unrotated*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
unrotated	“string”	–
variable	“string”	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Colloc_Rotated For Cvfem_Mesh_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Unrotated = *unrotated*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
unrotated	"string"	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Bc Colloc_Rotated For Cvfem_Mesh_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Unrotated = *unrotated*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
unrotated	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc Colloc_Rotated For Cvfem_Mesh_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Unrotated = *unrotated*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
unrotated	"string"	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Colloc_Rotated For Cvfem_Mesh_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Unrotated = *unrotated*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
unrotated	“string”	–
f	“string”	–

User_Field

Syntax

Bc Colloc_Rotated For Cvfem_Mesh_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Unrotated = *unrotated*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
unrotated	“string”	–
name	“string”	–
scaling	real	1
global_var	“string”	–
time_function	“string”	–

User_Function

Syntax

Bc Colloc_Rotated For Cvfem_Mesh_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Unrotated = *unrotated*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
unrotated	“string”	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Bc Colloc_Rotated For Cvfem_Mesh_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Unrotated = *unrotated*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
unrotated	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

Cvfem_Momentum

Constant

Syntax

Bc Colloc_Rotated For Cvfem_Momentum ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Unrotated = *unrotated*] | [Valuex = *valuex*] | [Valuey = *valuey*] | [Valuez = *valuez*])

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
unrotated	"string"	–
valuex	real	0
valuey	real	0
valuez	real	0

Kinematic

Syntax

Bc Colloc_Rotated For Cvfem_Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Kinematic ([Unrotated = *unrotated*] | [V0 = *v0*] | [Shared_Normal = *shared_normal*] | [No_Mesh_Movement = *no_mesh_movement*])

Summary

Kinematic model for rotated cvfem momentum

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
unrotated	“string”	–
v0	real	0
shared_normal	“string”	–
no_mesh_movement	“string”	–

Vector_String_Function

Syntax

Bc Colloc_Rotated For Cvfem_Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Vector_String_Function ([Unrotated = *unrotated*] | F_X = *f_x* | F_Y = *f_y* | [F_Z = *f_z*])

Summary

Value from a string function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
unrotated	“string”	–
f_x	“string”	–
f_y	“string”	–
f_z	“string”	–

Cvfem_Momentum_X

Constant

Syntax

Bc Colloc_Rotated For Cvfem_Momentum_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Unrotated = *unrotated*] | Value = *value*)

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
unrotated	“string”	–
value	real	–

Exponential

Syntax

Bc Colloc_Rotated For Cvfem_Momentum_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Unrotated = *unrotated*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
unrotated	“string”	–
variable	“string”	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Colloc_Rotated For Cvfem_Momentum_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Unrotated = *unrotated*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
unrotated	"string"	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Bc Colloc_Rotated For Cvfem_Momentum_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Unrotated = *unrotated*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
unrotated	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc Colloc_Rotated For Cvfem_Momentum_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Unrotated = *unrotated*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
unrotated	"string"	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Colloc_Rotated For Cvfem_Momentum_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Unrotated = *unrotated*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
unrotated	“string”	–
f	“string”	–

User_Field

Syntax

Bc Colloc_Rotated For Cvfem_Momentum_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Unrotated = *unrotated*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
unrotated	“string”	–
name	“string”	–
scaling	real	1
global_var	“string”	–
time_function	“string”	–

User_Function

Syntax

Bc Colloc_Rotated For Cvfem_Momentum_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Unrotated = *unrotated*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
unrotated	“string”	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Bc Colloc_Rotated For Cvfem_Momentum_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Unrotated = *unrotated*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
unrotated	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

Cvfem_Momentum_Y

Constant

Syntax

Bc Colloc_Rotated For Cvfem_Momentum_Y ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Unrotated = *unrotated*] | Value = *value*)

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
unrotated	"string"	–
value	real	–

Exponential

Syntax

Bc Colloc_Rotated For Cvfem_Momentum_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Unrotated = *unrotated*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
unrotated	“string”	–
variable	“string”	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Colloc_Rotated For Cvfem_Momentum_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Unrotated = *unrotated*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
unrotated	"string"	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Bc Colloc_Rotated For Cvfem_Momentum_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Unrotated = *unrotated*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
unrotated	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc Colloc_Rotated For Cvfem_Momentum_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Unrotated = *unrotated*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
unrotated	"string"	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Colloc_Rotated For Cvfem_Momentum_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Unrotated = *unrotated*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
unrotated	“string”	–
f	“string”	–

User_Field

Syntax

Bc Colloc_Rotated For Cvfem_Momentum_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Unrotated = *unrotated*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
unrotated	“string”	–
name	“string”	–
scaling	real	1
global_var	“string”	–
time_function	“string”	–

User_Function

Syntax

Bc Colloc_Rotated For Cvfem_Momentum_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Unrotated = *unrotated*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
unrotated	“string”	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Bc Colloc_Rotated For Cvfem_Momentum_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Unrotated = *unrotated*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
unrotated	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

Cvfem_Momentum_Z

Constant

Syntax

Bc Colloc_Rotated For Cvfem_Momentum_Z ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Unrotated = *unrotated*] | Value = *value*)

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
unrotated	"string"	–
value	real	–

Exponential

Syntax

Bc Colloc_Rotated For Cvfem_Momentum_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Unrotated = *unrotated*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
unrotated	“string”	–
variable	“string”	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Colloc_Rotated For Cvfem_Momentum_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Unrotated = *unrotated*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
unrotated	"string"	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Bc Colloc_Rotated For Cvfem_Momentum_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Unrotated = *unrotated*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
unrotated	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc Colloc_Rotated For Cvfem_Momentum_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Unrotated = *unrotated*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
unrotated	"string"	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Colloc_Rotated For Cvfem_Momentum_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Unrotated = *unrotated*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
unrotated	“string”	–
f	“string”	–

User_Field

Syntax

Bc Colloc_Rotated For Cvfem_Momentum_Z ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Unrotated = *unrotated*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
unrotated	“string”	–
name	“string”	–
scaling	real	1
global_var	“string”	–
time_function	“string”	–

User_Function

Syntax

Bc Colloc_Rotated For Cvfem_Momentum_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Unrotated = *unrotated*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
unrotated	“string”	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Bc Colloc_Rotated For Cvfem_Momentum_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Unrotated = *unrotated*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
unrotated	“string”	–
Name	“string”	–
plugin_parameters	“string”	–

Mesh

Constant

Syntax

Bc Colloc_Rotated For Mesh ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Unrotated = *unrotated*] | Value = *value*)

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
unrotated	“string”	–
value	real	–

Exponential

Syntax

Bc Colloc_Rotated For Mesh ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Unrotated = *unrotated*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
unrotated	“string”	–
variable	“string”	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Colloc_Rotated For Mesh ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Unrotated = *unrotated*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
unrotated	"string"	–
global_name	"string"	–
component	integer	0

Kinematic

Syntax

Bc Colloc_Rotated For Mesh ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Kinematic ([Unrotated = *unrotated*] | [V0 = *v0*] | [Shared_Normal = *shared_normal*] | [No_Mesh_Movement = *no_mesh_movement*])

Summary

Kinematic model for rotated mesh boundary condition

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
unrotated	"string"	–
v0	real	0
shared_normal	"string"	–
no_mesh_movement	"string"	–

Nd_Table

Syntax

Bc Colloc_Rotated For Mesh ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Unrotated = *unrotated*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
unrotated	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc Colloc_Rotated For Mesh ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Unrotated = *unrotated*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
unrotated	“string”	–
variable	“string”	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Pyrolysis

Syntax

Bc Colloc_Rotated For Mesh ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Pyrolysis ([Unrotated = *unrotated*] | [T_Ref = *t_ref*] | K = *k* | K_0 = *k_0* | N = *n* | [Temperature = *temperature*])

Summary

Pyrolysis model for rotated mesh boundary condition

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
unrotated	“string”	–
t_ref	real	0
k	real	–
k_0	real	–
n	real	–
temperature	“string”	–

Recession

Syntax

Bc Colloc_Rotated For Mesh ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Recession ([Unrotated = *unrotated*] | [P_Ref = *p_ref*] | K = *k* | K_0 = *k_0* | N = *n* | [Pressure = *pressure*])

Summary

Recession model for rotated mesh boundary condition

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
unrotated	"string"	–
p_ref	real	0
k	real	–
k_0	real	–
n	real	–
pressure	"string"	–

Scalar_String_Function

Syntax

Bc Colloc_Rotated For Mesh ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Unrotated = *unrotated*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
unrotated	“string”	–
f	“string”	–

User_Field

Syntax

Bc Colloc_Rotated For Mesh ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Unrotated = *unrotated*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
unrotated	“string”	–
name	“string”	–
scaling	real	1
global_var	“string”	–
time_function	“string”	–

User_Function

Syntax

Bc Colloc_Rotated For Mesh ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Unrotated = *unrotated*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
unrotated	“string”	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Bc Colloc_Rotated For Mesh ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Unrotated = *unrotated*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
unrotated	“string”	–
Name	“string”	–
plugin_parameters	“string”	–

Momentum

Constant

Syntax

Bc Colloc_Rotated For Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Unrotated = *unrotated*] | [Valuex = *valuex*] | [Valuey = *valuey*] | [Valuez = *valuez*])

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
unrotated	“string”	–
valuex	real	0
valuey	real	0
valuez	real	0

Kinematic

Syntax

Bc Colloc_Rotated For Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Kinematic ([Unrotated = *unrotated*] | [V0 = *v0*] | [Shared_Normal = *shared_normal*] | [No_Mesh_Movement = *no_mesh_movement*])

Summary

Kinematic model for rotated momentum

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
unrotated	“string”	–
v0	real	0
shared_normal	“string”	–
no_mesh_movement	“string”	–

Vector_String_Function

Syntax

Bc Colloc_Rotated For Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Vector_String_Function ([Unrotated = *unrotated*] | F_X = *f_x* | F_Y = *f_y* | [F_Z = *f_z*])

Summary

Value from a string function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
unrotated	"string"	–
f_x	"string"	–
f_y	"string"	–
f_z	"string"	–

Momentum_X

Constant

Syntax

Bc Colloc_Rotated For Momentum_X ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Unrotated = *unrotated*] | Value = *value*)

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
unrotated	"string"	–
value	real	–

Exponential

Syntax

Bc Colloc_Rotated For Momentum_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Unrotated = *unrotated*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
unrotated	“string”	–
variable	“string”	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Colloc_Rotated For Momentum_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Unrotated = *unrotated*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
unrotated	"string"	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Bc Colloc_Rotated For Momentum_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Unrotated = *unrotated*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
unrotated	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc Colloc_Rotated For Momentum_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Unrotated = *unrotated*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
unrotated	"string"	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Colloc_Rotated For Momentum_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Unrotated = *unrotated*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
unrotated	“string”	–
f	“string”	–

User_Field

Syntax

Bc Colloc_Rotated For Momentum_X ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Unrotated = *unrotated*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
unrotated	“string”	–
name	“string”	–
scaling	real	1
global_var	“string”	–
time_function	“string”	–

User_Function

Syntax

Bc Colloc_Rotated For Momentum_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Unrotated = *unrotated*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
unrotated	“string”	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Bc Colloc_Rotated For Momentum_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Unrotated = *unrotated*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
unrotated	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

Momentum_Y

Constant

Syntax

Bc Colloc_Rotated For Momentum_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Unrotated = *unrotated*] | Value = *value*)

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
unrotated	"string"	–
value	real	–

Exponential

Syntax

Bc Colloc_Rotated For Momentum_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Unrotated = *unrotated*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
unrotated	“string”	–
variable	“string”	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Colloc_Rotated For Momentum_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Unrotated = *unrotated*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
unrotated	"string"	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Bc Colloc_Rotated For Momentum_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Unrotated = *unrotated*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
unrotated	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc Colloc_Rotated For Momentum_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Unrotated = *unrotated*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
unrotated	"string"	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Colloc_Rotated For Momentum_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Unrotated = *unrotated*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
unrotated	“string”	–
f	“string”	–

User_Field

Syntax

Bc Colloc_Rotated For Momentum_Y ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Unrotated = *unrotated*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
unrotated	“string”	–
name	“string”	–
scaling	real	1
global_var	“string”	–
time_function	“string”	–

User_Function

Syntax

Bc Colloc_Rotated For Momentum_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Unrotated = *unrotated*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
unrotated	“string”	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Bc Colloc_Rotated For Momentum_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Unrotated = *unrotated*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
unrotated	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

Momentum_Z

Constant

Syntax

Bc Colloc_Rotated For Momentum_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Unrotated = *unrotated*] | Value = *value*)

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
unrotated	"string"	–
value	real	–

Exponential

Syntax

Bc Colloc_Rotated For Momentum_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Unrotated = *unrotated*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
unrotated	“string”	–
variable	“string”	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Colloc_Rotated For Momentum_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Unrotated = *unrotated*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
unrotated	"string"	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Bc Colloc_Rotated For Momentum_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Unrotated = *unrotated*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
unrotated	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc Colloc_Rotated For Momentum_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Unrotated = *unrotated*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
unrotated	"string"	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Colloc_Rotated For Momentum_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Unrotated = *unrotated*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
unrotated	“string”	–
f	“string”	–

User_Field

Syntax

Bc Colloc_Rotated For Momentum_Z ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Unrotated = *unrotated*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
unrotated	“string”	–
name	“string”	–
scaling	real	1
global_var	“string”	–
time_function	“string”	–

User_Function

Syntax

Bc Colloc_Rotated For Momentum_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Unrotated = *unrotated*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
unrotated	“string”	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Bc Colloc_Rotated For Momentum_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Unrotated = *unrotated*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
unrotated	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

Bc Diffuse

Ampere

Constant

Syntax

Bc Diffuse For Ampere ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | [Valuex = *valuex*] | [Valuey = *valuey*] | [Valuez = *valuez*])

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
valuex	real	0
valuey	real	0
valuez	real	0

Vector_String_Function

Syntax

Bc Diffuse For Ampere ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Vector_String_Function ([Toggle = *toggle*] | F_X = *f_x* | F_Y = *f_y* | [F_Z = *f_z*])

Summary

Value from a string function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f_x	"string"	–
f_y	"string"	–
f_z	"string"	–

Ampere_X

Constant

Syntax

Bc Diffuse For Ampere_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | Value = *value*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
value	real	–

Exponential

Syntax

Bc Diffuse For Ampere_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Diffuse For Ampere_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Bc Diffuse For Ampere_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc Diffuse For Ampere_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Diffuse For Ampere_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f	"string"	–

User_Field

Syntax

Bc Diffuse For Ampere_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc Diffuse For Ampere_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc Diffuse For Ampere_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

Ampere_Y

Constant

Syntax

Bc Diffuse For Ampere_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | Value = *value*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
value	real	–

Exponential

Syntax

Bc Diffuse For Ampere_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Diffuse For Ampere_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Bc Diffuse For Ampere_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc Diffuse For Ampere_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Diffuse For Ampere_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f	"string"	–

User_Field

Syntax

Bc Diffuse For Ampere_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc Diffuse For Ampere_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc Diffuse For Ampere_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

Ampere_Z

Constant

Syntax

Bc Diffuse For Ampere_Z ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | Value = *value*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
value	real	–

Exponential

Syntax

Bc Diffuse For Ampere_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Diffuse For Ampere_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Bc Diffuse For Ampere_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc Diffuse For Ampere_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Diffuse For Ampere_Z ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f	"string"	–

User_Field

Syntax

Bc Diffuse For Ampere_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc Diffuse For Ampere_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc Diffuse For Ampere_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

Charge_Density

Constant

Syntax

Bc Diffuse For Charge_Density ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | *{flux | value}* = *flux*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
flux	real	–

Exponential

Syntax

Bc Diffuse For Charge_Density ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Diffuse For Charge_Density ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
global_name	"string"	–
component	integer	0

Nat_E

Syntax

Bc Diffuse For Charge_Density ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nat_E ([Toggle = *toggle*] | [Sigma = *sigma*])

Summary

This model is now deprecated

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
sigma	"string"	–

Nd_Table

Syntax

Bc Diffuse For Charge_Density ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc Diffuse For Charge_Density ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
variable	“string”	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Diffuse For Charge_Density ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
f	“string”	–

User_Field

Syntax

Bc Diffuse For Charge_Density ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
scaling	real	1
global_var	“string”	–
time_function	“string”	–

User_Function

Syntax

Bc Diffuse For Charge_Density ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc Diffuse For Charge_Density ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

Continuity

Constant

Syntax

Bc Diffuse For Continuity ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | Value = *value*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
value	real	–

Exponential

Syntax

Bc Diffuse For Continuity ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Extrapolated_Pressure

Syntax

Bc Diffuse For Continuity ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Extrapolated_Pressure [Toggle = *toggle*]

Summary

This model has been deprecated and removed from the code.

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–

Global

Syntax

Bc Diffuse For Continuity ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
global_name	“string”	–
component	integer	0

Nd_Table

Syntax

Bc Diffuse For Continuity ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc Diffuse For Continuity ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
variable	“string”	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Pspg

Syntax

Bc Diffuse For Continuity ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Pspg ([Toggle = *toggle*] | [Tausurfacefactor = *TauSurfaceFactor*] | [Tausurfacejacobianfactor = *TauSurfaceJacobianFactor*])

Summary

Continuity PSPG flux BC

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
TauSurfaceFactor	real	1
TauSurfaceJacobianFactor	real	1

Scalar_String_Function

Syntax

Bc Diffuse For Continuity ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f	"string"	–

User_Field

Syntax

Bc Diffuse For Continuity ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc Diffuse For Continuity ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc Diffuse For Continuity ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

Current

Calore_User_Sub

Syntax

Bc Diffuse For Current ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Calore_User_Sub ([Toggle = *toggle*] | Name = *name* | Type = *type* | [Multiplier = *multiplier*] | [Material_Data_Block = *material_data_block*] | [Data = *data*] | [Scaling_Field = *scaling_field*])

Summary

Values from a Calore user subroutine

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
type	“string”	–
multiplier	real	1
material_data_block	“string”	–
data	“string”	–
scaling_field	“string”	–

Constant

Syntax

Bc Diffuse For Current ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | *{flux | value}* = *flux*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
flux	real	–

Constant_With_Cutoff_Voltage

Syntax

Bc Diffuse For Current ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant_With_Cutoff_Voltage ([Toggle = *toggle*] | Value = *value* | Cutoff_Value = *cutoff_value*)

Summary

Apply a constant current until a cutoff voltage is crossed anywhere on the surface. After that apply 0 current.

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
value	real	–
cutoff_value	real	–

Exponential

Syntax

Bc Diffuse For Current ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Diffuse For Current ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
global_name	“string”	–
component	integer	0

Nd_Table

Syntax

Bc Diffuse For Current ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc Diffuse For Current ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Resistive_Load

Syntax

Bc Diffuse For Current ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Resistive_Load ([Toggle = *toggle*] | [Area = *Area*] | [Referencevoltage = *ReferenceVoltage*])

Summary

Model a boundary attached to a resistive load. The local applied current density is V / (R A) where V is the local voltage, R is the specified resistance, and A is the area of the surface the BC is applied to.

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Area	real	-1
ReferenceVoltage	real	0

Scalar_String_Function

Syntax

Bc Diffuse For Current ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f	"string"	–

User_Field

Syntax

Bc Diffuse For Current ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc Diffuse For Current ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc Diffuse For Current ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

Cvfem_Continuity

Constant

Syntax

Bc Diffuse For Cvfem_Continuity ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | Value = *value*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
value	real	–

Exponential

Syntax

Bc Diffuse For Cvfem_Continuity ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Diffuse For Cvfem_Continuity ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
global_name	"string"	–
component	integer	0

Inflow

Syntax

Bc Diffuse For Cvfem_Continuity ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Inflow [Toggle = *toggle*]

Summary

IBP continuity inflow; nodal interp

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–

Interface

Syntax

Bc Diffuse For Cvfem_Continuity ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Interface [Toggle = *toggle*]

Summary

INTERFACE CVFEM continuity flux

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–

Mass_Flux

Syntax

Bc Diffuse For Cvfem_Continuity ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Mass_Flux ([Toggle = *toggle*] | Total_Mdot = *total_mdot*)

Summary

Mass flux CVFEM continuity flux

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
total_mdot	real	–

Nd_Table

Syntax

Bc Diffuse For Cvfem_Continuity ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Open_Flow

Syntax

Bc Diffuse For Cvfem_Continuity ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Open_Flow ([Toggle = *toggle*] | [Pressure = *pressure*] | [Total_Pressure = *total_pressure*])

Summary

Open bc for continuity

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
pressure	real	0
total_pressure	real	0

Polynomial

Syntax

Bc Diffuse For Cvfem_Continuity ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Porous_Robin_Averaged_Coeff

Syntax

Bc Diffuse For Cvfem_Continuity ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Porous_Robin_Averaged_Coeff ([Toggle = *toggle*] | P_Field = *P_Field* | Massflux_Field = *MassFlux_Field* | Oppositecoeff_Field = *OppositeCoeff_Field*)

Summary

Boundary condition for loose coupling of continuity to a free fluid region

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
P_Field	"string"	–
MassFlux_Field	"string"	–
OppositeCoeff_Field	"string"	–

Porous_Robin_One_Region

Syntax

Bc Diffuse For Cvfem_Continuity ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Porous_Robin_One_Region [Toggle = *toggle*]

Summary

Boundary condition for loose coupling of continuity to a free fluid region

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–

Scalar_String_Function

Syntax

Bc Diffuse For Cvfem_Continuity ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
f	“string”	–

Simple_Inflow

Syntax

Bc Diffuse For Cvfem_Continuity ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Simple_Inflow ([Toggle = *toggle*] | *{density | value}* = *Density* | [Velocity_X = *Velocity_X*] | [Velocity_Y = *Velocity_Y*] | [Velocity_Z = *Velocity_Z*])

Summary

IBP continuity inflow; specified dens and u

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Density	real	–
Velocity_X	real	0
Velocity_Y	real	0
Velocity_Z	real	0

User_Field

Syntax

Bc Diffuse For Cvfem_Continuity ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc Diffuse For Cvfem_Continuity ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc Diffuse For Cvfem_Continuity ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

Cvfem_Density_Edge_Gradient_Projection

Constant

Syntax

Bc Diffuse For Cvfem_Density_Edge_Gradient_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | Value = *value*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
value	real	–

Exponential

Syntax

Bc Diffuse For Cvfem_Density_Edge_Gradient_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Diffuse For Cvfem_Density_Edge_Gradient_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Bc Diffuse For Cvfem_Density_Edge_Gradient_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc Diffuse For Cvfem_Density_Edge_Gradient_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Diffuse For Cvfem_Density_Edge_Gradient_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f	"string"	–

Simple_Interp

Syntax

Bc Diffuse For Cvfem_Density_Edge_Gradient_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Simple_Interp [Toggle = *toggle*]

Summary

CVFEM edge-based projection simple interpolation flux BC

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–

User_Field

Syntax

Bc Diffuse For Cvfem_Density_Edge_Gradient_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
scaling	real	1
global_var	“string”	–
time_function	“string”	–

User_Function

Syntax

Bc Diffuse For Cvfem_Density_Edge_Gradient_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc Diffuse For Cvfem_Density_Edge_Gradient_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

Cvfem_Dispersed_Continuity

Constant

Syntax

Bc Diffuse For Cvfem_Dispersed_Continuity ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | Value = *value*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
value	real	–

Exponential

Syntax

Bc Diffuse For Cvfem_Dispersed_Continuity ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Diffuse For Cvfem_Dispersed_Continuity ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
global_name	"string"	–
component	integer	0

Inflow

Syntax

Bc Diffuse For Cvfem_Dispersed_Continuity ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Inflow [Toggle = *toggle*]

Summary

Inflow mass flux for dispersed phase continuity equation; nodal interpolation

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–

Nd_Table

Syntax

Bc Diffuse For Cvfem_Dispersed_Continuity ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Open_Flow

Syntax

Bc Diffuse For Cvfem_Dispersed_Continuity ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Open_Flow [Toggle = *toggle*]

Summary

Open bc for dispersed continuity

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–

Polynomial

Syntax

Bc Diffuse For Cvfem_Dispersed_Continuity ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset,

X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Diffuse For Cvfem_Dispersed_Continuity ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be

a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f	"string"	–

Simple_Inflow

Syntax

Bc Diffuse For Cvfem_Dispersed_Continuity ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Simple_Inflow ([Toggle = *toggle*] | [Velocity_X = *Velocity_X*] | [Velocity_Y = *Velocity_Y*] | [Velocity_Z = *Velocity_Z*])

Summary

Inflow mass flux for dispersed continuity equation; specified density and velocity

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Velocity_X	real	0
Velocity_Y	real	0
Velocity_Z	real	0

User_Field

Syntax

Bc Diffuse For Cvfem_Dispersed_Continuity ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc Diffuse For Cvfem_Dispersed_Continuity ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc Diffuse For Cvfem_Dispersed_Continuity ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

Cvfem_Dispersed_Momentum

Constant

Syntax

Bc Diffuse For Cvfem_Dispersed_Momentum ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | [Valuex = *valuex*] | [Valuey = *valuey*] | [Valuez = *valuez*])

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
valuex	real	0
valuey	real	0
valuez	real	0

Open_Adv_Flow

Syntax

Bc Diffuse For Cvfem_Dispersed_Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Open_Adv_Flow ([Toggle = *toggle*] | [Coflow_Entrainment_Valuex = *Coflow_Entrainment_ValueX*] | [Coflow_Entrainment_Valuey = *Coflow_Entrainment_ValueY*] | [Coflow_Entrainment_Valuez = *Coflow_Entrainment_ValueZ*])

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Coflow_Entrainment_ValueX	real	0
Coflow_Entrainment_ValueY	real	0
Coflow_Entrainment_ValueZ	real	0

Open_Flow

Syntax

Bc Diffuse For Cvfem_Dispersed_Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Open_Flow [Toggle = *toggle*]

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–

Symmetry_Flow

Syntax

Bc Diffuse For Cvfem_Dispersed_Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Symmetry_Flow [Toggle = *toggle*]

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–

Vector_String_Function

Syntax

Bc Diffuse For Cvfem_Dispersed_Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Vector_String_Function ([Toggle = *toggle*] | F_X = *f_x* | F_Y = *f_y* | [F_Z = *f_z*])

Summary

Value from a string function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f_x	"string"	–
f_y	"string"	–
f_z	"string"	–

Cvfem_Dispersed_Momentum_X

Constant

Syntax

Bc Diffuse For Cvfem_Dispersed_Momentum_X ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | Value = *value*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
value	real	–

Exponential

Syntax

Bc Diffuse For Cvfem_Dispersed_Momentum_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Diffuse For Cvfem_Dispersed_Momentum_X ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Bc Diffuse For Cvfem_Dispersed_Momentum_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc Diffuse For Cvfem_Dispersed_Momentum_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Diffuse For Cvfem_Dispersed_Momentum_X ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f	"string"	–

User_Field

Syntax

Bc Diffuse For Cvfem_Dispersed_Momentum_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc Diffuse For Cvfem_Dispersed_Momentum_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc Diffuse For Cvfem_Dispersed_Momentum_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

Cvfem_Dispersed_Momentum_Y

Constant

Syntax

Bc Diffuse For Cvfem_Dispersed_Momentum_Y ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | Value = *value*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
value	real	–

Exponential

Syntax

Bc Diffuse For Cvfem_Dispersed_Momentum_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Diffuse For Cvfem_Dispersed_Momentum_Y ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Bc Diffuse For Cvfem_Dispersed_Momentum_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc Diffuse For Cvfem_Dispersed_Momentum_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Diffuse For Cvfem_Dispersed_Momentum_Y ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f	"string"	–

User_Field

Syntax

Bc Diffuse For Cvfem_Dispersed_Momentum_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc Diffuse For Cvfem_Dispersed_Momentum_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc Diffuse For Cvfem_Dispersed_Momentum_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

Cvfem_Dispersed_Momentum_Z

Constant

Syntax

Bc Diffuse For Cvfem_Dispersed_Momentum_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | Value = *value*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
value	real	–

Exponential

Syntax

Bc Diffuse For Cvfem_Dispersed_Momentum_Z ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Diffuse For Cvfem_Dispersed_Momentum_Z ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Bc Diffuse For Cvfem_Dispersed_Momentum_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc Diffuse For Cvfem_Dispersed_Momentum_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Diffuse For Cvfem_Dispersed_Momentum_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f	"string"	–

User_Field

Syntax

Bc Diffuse For Cvfem_Dispersed_Momentum_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc Diffuse For Cvfem_Dispersed_Momentum_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc Diffuse For Cvfem_Dispersed_Momentum_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

Cvfem_Div_Projection

Constant

Syntax

Bc Diffuse For Cvfem_Div_Projection ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | [Valuexx = *valuexx*] | [Valuexy = *valuexy*] | [Valuexz = *valuexz*] | [Valueyx = *valueyx*] | [Valueyy = *valueyy*] | [Valueyz = *valueyz*] | [Valuezx = *valuezx*] | [Valuezy = *valuezy*] | [Valuezz = *valuezz*] | [Valuet11 = *valuet11*] | [Valuet12 = *valuet12*] | [Valuet13 = *valuet13*] | [Valuet21 = *valuet21*] | [Valuet22 = *valuet22*] | [Valuet23 = *valuet23*] | [Valuet31 = *valuet31*] | [Valuet32 = *valuet32*] | [Valuet33 = *valuet33*])

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
valuexx	real	0
valuexy	real	0
valuexz	real	0
valueyx	real	0
valueyy	real	0
valueyz	real	0
valuezx	real	0
valuezy	real	0
valuezz	real	0
valuet11	real	0
valuet12	real	0
valuet13	real	0
valuet21	real	0
valuet22	real	0
valuet23	real	0
valuet31	real	0
valuet32	real	0
valuet33	real	0

Simple_Interp_Tensor

Syntax

Bc Diffuse For Cvfem_Div_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Simple_Interp_Tensor ([Toggle = *toggle*] | Source = *source*)

Summary

CVFEM divergence projection simple interpolation tensor flux BC

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
source	"string"	–

Simple_Interp_Vector

Syntax

Bc Diffuse For Cvfem_Div_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Simple_Interp_Vector ([Toggle = *toggle*] | Source = *source*)

Summary

CVFEM divergence projection simple interpolation vector flux BC

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
source	"string"	–

Tensor_String_Function

Syntax

Bc Diffuse For Cvfem_Div_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Tensor_String_Function ([Toggle = *toggle*] | F_Xx = *f_xx* | F_Yy = *f_yy* | [F_Xy = *f_xy*] | [F_Zz = *f_zz*] | [F_Xz = *f_xz*] | [F_Yz = *f_yz*] | [F_Yx = *f_yx*] | [F_Zx = *f_zx*] | [F_Zy = *f_zy*])

Summary

Value from a string function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f_xx	"string"	–
f_yy	"string"	–
f_xy	"string"	0.0
f_zz	"string"	–
f_xz	"string"	0.0
f_yz	"string"	0.0
f_yx	"string"	0.0
f_zx	"string"	0.0
f_zy	"string"	0.0

Tensor_String_Function_Symmetric

Syntax

Bc Diffuse For Cvfem_Div_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Tensor_String_Function_Symmetric ([Toggle = *toggle*] | F_Xx = *f_xx* | F_Yy = *f_yy* | [F_Xy = *f_xy*] | [F_Zz = *f_zz*] | [F_Xz = *f_xz*] | [F_Yz = *f_yz*])

Summary

Value from a string function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
f_xx	“string”	–
f_yy	“string”	–
f_xy	“string”	0.0
f_zz	“string”	–
f_xz	“string”	0.0
f_yz	“string”	0.0

Cvfem_Energy

Constant

Syntax

Bc Diffuse For Cvfem_Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Data = *data*] | *{flux | value}* = *flux*)

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
data	“string”	–
flux	real	–

Exponential

Syntax

Bc Diffuse For Cvfem_Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Data = *data*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
data	“string”	–
variable	“string”	–
constant	real	0
multiplier	real	1
exponent	real	–

Exponential_Vapor_Cooling

Syntax

Bc Diffuse For Cvfem_Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential_Vapor_Cooling ([Data = *data*] | Tboil = *tboil* | L = *l* | Pa = *pa* | Mw = *mw* | Alpha = *alpha* | [Tl = *tl*] | [Th = *th*])

Summary

Exponential vapor cooling energy flux

Description

Flux is given by:

$$F = P_a L\alpha \exp\left(\frac{L}{R}\left(\frac{1}{T_{boil}} - \frac{1}{T}\right)\right)\sqrt{\frac{1}{2\pi RT}}$$

where R is the specific gas constant ($\bar{R}$/MW). For T less than T_L, zero flux is applied. For T between T_L and T_H, a 3rd order smoothing polynomial is used to provide continuity of values and derivatives between 0 and the expression given above. For T greater than T_H, the above expression is used. If not provided, T_L and T_H are set to 10% below and above T_{Boil}, respectively.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
data	“string”	–
tboil	real	–
l	real	–
pa	real	–
mw	real	–
alpha	real	–
tl	real	-1
th	real	-1

Generalized_Nat_Conv

Syntax

Bc Diffuse For Cvfem_Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Generalized_Nat_Conv ([Data = *data*] | [Multiplier = *multiplier*] | [H = *h*] | [T_Ref = *t_ref*])

Summary

CVFEM energy generalized natural convection flux BC

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
data	"string"	–
multiplier	real	1
h	"string"	–
t_ref	"string"	–

Generalized_Rad

Syntax

Bc Diffuse For Cvfem_Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Generalized_Rad ([Data = *data*] | [Multiplier = *multiplier*] | [T_Ref = *t_ref*])

Summary

Radiative energy flux, with values for emissivity, form factor and irradiation taken from expressions

Description

Flux is:

$$M\epsilon f\left(\sigma T^4 - I\right)$$

M is MULTIPLIER. σ is the Stefan-Boltzmann constant. ϵ and F are expressions for emissivity and form factor. T is surface temperature. I is the expression for irradiation. If I does not already exist, it will be constructed as σT_{ref}^4, where T_{ref} is either provided by the BC Rad Reference Temperature expression or the inline T_{ref} parameter (constant or string function).

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
data	“string”	–
multiplier	real	1
t_ref	“string”	–

Global

Syntax

Bc Diffuse For Cvfem_Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Data = *data*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
data	“string”	–
global_name	“string”	–
component	integer	0

Interface

Syntax

Bc Diffuse For Cvfem_Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Interface [Data = *data*]

Summary

INTERFACE CVFEM energy flux

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
data	“string”	–

Mass_Flux

Syntax

Bc Diffuse For Cvfem_Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Mass_Flux ([Data = *data*] | Inflow_Phi = *Inflow_Phi* | Total_Mdot = *total_mdot*)

Summary

CVFEM energy mass flux BC

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
data	"string"	–
Inflow_Phi	real	–
total_mdot	real	–

Mass_Flux_From_Temperature

Syntax

Bc Diffuse For Cvfem_Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Mass_Flux_From_Temperature ([Data = *data*] | [Temperature = *temperature*] | [Pressure = *pressure*] | [Mass_Fraction_0 = *mass_fraction_0*] | [Mass_Fraction_1 = *mass_fraction_1*] | [Mass_Fraction_2 = *mass_fraction_2*] | [Mass_Fraction_3 = *mass_fraction_3*] | [Mass_Fraction_4 = *mass_fraction_4*] | [Mass_Fraction_5 = *mass_fraction_5*] | [Mass_Fraction_6 = *mass_fraction_6*] | [Mass_Fraction_7 = *mass_fraction_7*] | [Mass_Fraction_8 = *mass_fraction_8*] | Total_Mdot = *total_mdot*)

Summary

CVFEM energy mass flux from temperature flux BC

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
data	“string”	–
temperature	real	0
pressure	real	0
mass_fraction_0	real	0
mass_fraction_1	real	0
mass_fraction_2	real	0
mass_fraction_3	real	0
mass_fraction_4	real	0
mass_fraction_5	real	0
mass_fraction_6	real	0
mass_fraction_7	real	0
mass_fraction_8	real	0
total_mdot	real	–

Nat_Conv

Syntax

Bc Diffuse For Cvfem_Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nat_Conv ([Data = *data*] | H = *h* | T_Ref = *t_ref*)

Summary

CVFEM convective energy flux defined with constant models for heat transfer coefficient and reference temperature

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
data	"string"	–
h	real	–
t_ref	real	–

Nd_Table

Syntax

Bc Diffuse For Cvfem_Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Data = *data*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
data	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Open_Adv_Flow

Syntax

Bc Diffuse For Cvfem_Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Open_Adv_Flow ([Data = *data*] | Far_Field_Entrainment_Value = *Far_Field_Entrainment_Value* | [Pressure = *pressure*] | [Total_Pressure = *total_pressure*])

Summary

CVFEM energy open advective flow flux

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
data	"string"	–
Far_Field_Entrainment_Value	real	–
pressure	real	0
total_pressure	real	0

Open_Adv_Flow_From_Temperature

Syntax

Bc Diffuse For Cvfem_Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Open_Adv_Flow_From_Temperature ([Data = *data*] | [Temperature = *temperature*] | [Pressure = *pressure*] | [Mass_Fraction_0 = *mass_fraction_0*] | [Mass_Fraction_1 = *mass_fraction_1*] | [Mass_Fraction_2 = *mass_fraction_2*] | [Mass_Fraction_3 = *mass_fraction_3*] | [Mass_Fraction_4 = *mass_fraction_4*] | [Mass_Fraction_5 = *mass_fraction_5*] | [Mass_Fraction_6 = *mass_fraction_6*] | [Mass_Fraction_7 = *mass_fraction_7*] | [Mass_Fraction_8 = *mass_fraction_8*] | [Total_Pressure = *total_pressure*])

Summary

CVFEM energy open advective flow from temperature flux

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
data	“string”	–
temperature	real	0
pressure	real	0
mass_fraction_0	real	0
mass_fraction_1	real	0
mass_fraction_2	real	0
mass_fraction_3	real	0
mass_fraction_4	real	0
mass_fraction_5	real	0
mass_fraction_6	real	0
mass_fraction_7	real	0
mass_fraction_8	real	0
total_pressure	real	0

Polynomial

Syntax

Bc Diffuse For Cvfem_Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Data = *data*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
data	“string”	–
variable	“string”	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Porous_Robin_Coupled

Syntax

Bc Diffuse For Cvfem_Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Porous_Robin_Coupled ([Data = *data*] | Enthalpy_Field = *enthalpy_field* | Enthalpy_Diffusive_Flux_Field = *enthalpy_diffusive_flux_field* | Averaged_Coeff_Field = *averaged_coeff_field*)

Summary

Boundary condition for loose coupling of enthalpy to a porous region

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
data	“string”	–
enthalpy_field	“string”	–
enthalpy_diffusive_flux_field	“string”	–
averaged_coeff_field	“string”	–

Porous_Robin_Coupled_One_Region

Syntax

Bc Diffuse For Cvfem_Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Porous_Robin_Coupled_One_Region [Data = *data*]

Summary

Boundary condition for loose coupling of enthalpy to a porous region

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
data	“string”	–

Porous_Robin_Coupled_With_Solid_Phase_Convection

Syntax

Bc Diffuse For Cvfem_Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Porous_Robin_Coupled_With_Solid_Phase_Convection ([Data = *data*] | [Multiplier = *multiplier*] | Enthalpy_Field = *enthalpy_field* | Enthalpy_Diffusive_Flux_Field = *enthalpy_diffusive_flux_field* | Solid_Phase_Temperature_Field = *solid_phase_temperature_field* | Porosity_Field = *porosity_field* | Averaged_Coeff_Field = *averaged_coeff_field* | Volumetric_Heat_Transfer_Coeff_Field = *volumetric_heat_transfer_coeff_field* | Specific_Surface_Area_Field = *specific_surface_area_field*)

Summary

Boundary condition for loose coupling of enthalpy to a porous region

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
data	“string”	–
multiplier	real	1
enthalpy_field	“string”	–
enthalpy_diffusive_flux_field	“string”	–
solid_phase_temperature_field	“string”	–
porosity_field	“string”	–
averaged_coeff_field	“string”	–
volumetric_heat_transfer_coeff_field	“string”	–
specific_surface_area_field	“string”	–

Porous_Robin_Coupled_With_Solid_Phase_Convection_One_Region

Syntax

Bc Diffuse For Cvfem_Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Porous_Robin_Coupled_With_Solid_Phase_Convection_One_Region ([Data = *data*] | [Multiplier = *multiplier*])

Summary

Boundary condition for coupling of enthalpy to a porous region in the same region.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
data	“string”	–
multiplier	real	1

Rad

Syntax

Bc Diffuse For Cvfem_Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Rad ([Data = *data*] | Crad = *crad* | T_Ref = *t_ref*)

Summary

Radiative energy flux defined with constant models for emissivity and reference temperature.

Description

The applied flux is given by:

$$c_{rad}\left(T^4 - T_{ref}^4\right)$$

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
data	"string"	–
crad	real	–
t_ref	real	–

Ray_Trace

Syntax

Bc Diffuse For Cvfem_Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Ray_Trace ([Data = *data*] | Ray_Source = *ray_source* | [Emission = *emission*])

Summary

Apply a ray tracing energy flux

Description

Adds an energy flux based on the intersection of rays defined in a ray tracing source block for the equation system with name given by the RAY_SOURCE parameter with the surface. The optional EMISSION parameter also imposes a σT^4 blackbody radiative emission term

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
data	"string"	–
ray_source	"string"	–
emission	"string"	FALSE

Scalar_String_Function

Syntax

Bc Diffuse For Cvfem_Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Data = *data*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
data	"string"	–
f	"string"	–

User_Field

Syntax

Bc Diffuse For Cvfem_Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Data = *data*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
data	"string"	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc Diffuse For Cvfem_Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Data = *data*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
data	“string”	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Bc Diffuse For Cvfem_Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Data = *data*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
data	“string”	–
Name	“string”	–
plugin_parameters	“string”	–

Wall_Function_From_Cht_Temperature

Syntax

Bc Diffuse For Cvfem_Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Wall_Function_From_Cht_Temperature ([Data = *data*] | Wall_Friction_Factor = *wall_friction_factor*)

Summary

CVFEM energy wall function from conjugate heat transfer temperature flux BC

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
data	“string”	–
wall_friction_factor	real	–

Wall_Function_From_Temperature

Syntax

Bc Diffuse For Cvfem_Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Wall_Function_From_Temperature ([Data = *data*] | *{t | value}* = *t* | Wall_Friction_Factor = *wall_friction_factor*)

Summary

CVFEM energy wall function from temperature flux BC

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
data	"string"	–
t	real	–
wall_friction_factor	real	–

Cvfem_Level_Set

Open_Adv_Flow

Syntax

Bc Diffuse For Cvfem_Level_Set ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Open_Adv_Flow ([Toggle = *toggle*] | [Pressure = *pressure*] | [Total_Pressure = *total_pressure*] | Far_Field_Entrainment_Value = *far_field_entrainment_value*)

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
pressure	real	0
total_pressure	real	0
far_field_entrainment_value	real	–

Open_Nc_Adv_Flow

Syntax

Bc Diffuse For Cvfem_Level_Set ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Open_Nc_Adv_Flow ([Toggle = *toggle*] | Far_Field_Entrainment_Value = *far_field_entrainment_value*)

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
far_field_entrainment_value	real	–

Cvfem_Lumped_Div_Projection

Constant

Syntax

Bc Diffuse For Cvfem_Lumped_Div_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | [Valuexx = *valuexx*] | [Valuexy = *valuexy*] | [Valuexz = *valuexz*] | [Valueyx = *valueyx*] | [Valueyy = *valueyy*] | [Valueyz = *valueyz*] | [Valuezx = *valuezx*] | [Valuezy = *valuezy*] | [Valuezz = *valuezz*] | [Valuet11 = *valuet11*] | [Valuet12 = *valuet12*] | [Valuet13 = *valuet13*] | [Valuet21 = *valuet21*] | [Valuet22 = *valuet22*] | [Valuet23 = *valuet23*] | [Valuet31 = *valuet31*] | [Valuet32 = *valuet32*] | [Valuet33 = *valuet33*])

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
valuexx	real	0
valuexy	real	0
valuexz	real	0
valueyx	real	0
valueyy	real	0
valueyz	real	0
valuezx	real	0
valuezy	real	0
valuezz	real	0
valuet11	real	0
valuet12	real	0
valuet13	real	0
valuet21	real	0
valuet22	real	0
valuet23	real	0
valuet31	real	0
valuet32	real	0
valuet33	real	0

Simple_Interp_Tensor

Syntax

Bc Diffuse For Cvfem_Lumped_Div_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Simple_Interp_Tensor ([Toggle = *toggle*] | Source = *source*)

Summary

CVFEM lumped divergence projection simple interpolation tensor flux BC

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
source	"string"	–

Simple_Interp_Vector

Syntax

Bc Diffuse For Cvfem_Lumped_Div_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Simple_Interp_Vector ([Toggle = *toggle*] | Source = *source*)

Summary

CVFEM lumped divergence projection simple interpolation vector flux BC

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
source	"string"	–

Tensor_String_Function

Syntax

Bc Diffuse For Cvfem_Lumped_Div_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Tensor_String_Function ([Toggle = *toggle*] | F_Xx = *f_xx* | F_Yy = *f_yy* | [F_Xy = *f_xy*] | [F_Zz = *f_zz*] | [F_Xz = *f_xz*] | [F_Yz = *f_yz*] | [F_Yx = *f_yx*] | [F_Zx = *f_zx*] | [F_Zy = *f_zy*])

Summary

Value from a string function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f_xx	"string"	–
f_yy	"string"	–
f_xy	"string"	0.0
f_zz	"string"	–
f_xz	"string"	0.0
f_yz	"string"	0.0
f_yx	"string"	0.0
f_zx	"string"	0.0
f_zy	"string"	0.0

Tensor_String_Function_Symmetric

Syntax

Bc Diffuse For Cvfem_Lumped_Div_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Tensor_String_Function_Symmetric ([Toggle = *toggle*] | F_Xx = *f_xx* | F_Yy = *f_yy* | [F_Xy = *f_xy*] | [F_Zz = *f_zz*] | [F_Xz = *f_xz*] | [F_Yz = *f_yz*])

Summary

Value from a string function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
f_xx	“string”	–
f_yy	“string”	–
f_xy	“string”	0.0
f_zz	“string”	–
f_xz	“string”	0.0
f_yz	“string”	0.0

Cvfem_Lumped_Projection

Constant

Syntax

Bc Diffuse For Cvfem_Lumped_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | Value = *value*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
value	real	–

Exponential

Syntax

Bc Diffuse For Cvfem_Lumped_Projection ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Diffuse For Cvfem_Lumped_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Bc Diffuse For Cvfem_Lumped_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Open

Syntax

Bc Diffuse For Cvfem_Lumped_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Open ([Toggle = *toggle*] | [Total_Pressure = *total_pressure*] | [Pressure = *pressure*])

Summary

CVFEM lumped projection open flux BC

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
total_pressure	real	0
pressure	real	0

Polynomial

Syntax

Bc Diffuse For Cvfem_Lumped_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Diffuse For Cvfem_Lumped_Projection ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f	"string"	–

Simple_Interp

Syntax

Bc Diffuse For Cvfem_Lumped_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Simple_Interp [Toggle = *toggle*]

Summary

CVFEM lumped projection simple interpolation flux BC

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–

User_Field

Syntax

Bc Diffuse For Cvfem_Lumped_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
scaling	real	1
global_var	“string”	–
time_function	“string”	–

User_Function

Syntax

Bc Diffuse For Cvfem_Lumped_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc Diffuse For Cvfem_Lumped_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

Cvfem_Mass_Fraction

Constant

Syntax

Bc Diffuse For Cvfem_Mass_Fraction ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | Value = *value*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
value	real	–

Exponential

Syntax

Bc Diffuse For Cvfem_Mass_Fraction ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Diffuse For Cvfem_Mass_Fraction ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
global_name	"string"	–
component	integer	0

Mass_Flux

Syntax

Bc Diffuse For Cvfem_Mass_Fraction ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Mass_Flux ([Toggle = *toggle*] | Total_Mdot = *total_mdot* | [Inflow_Phi = *inflow_phi*])

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
total_mdot	real	–
inflow_phi	real	0

Nd_Table

Syntax

Bc Diffuse For Cvfem_Mass_Fraction ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Open_Adv_Flow

Syntax

Bc Diffuse For Cvfem_Mass_Fraction ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Open_Adv_Flow ([Toggle = *toggle*] | [Pressure = *pressure*] | [Total_Pressure = *total_pressure*] | [Far_Field_Entrainment_Value = *far_field_entrainment_value*])

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
pressure	real	0
total_pressure	real	0
far_field_entrainment_value	real	0

Polynomial

Syntax

Bc Diffuse For Cvfem_Mass_Fraction ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
variable	“string”	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Porous_Robin_Coupled

Syntax

Bc Diffuse For Cvfem_Mass_Fraction ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Porous_Robin_Coupled ([Toggle = *toggle*] | Mass_Fraction_Field = *mass_fraction_field* | Mass_Fraction_Diffusive_Flux_Field = *mass_fraction_diffusive_flux_field* | Averaged_Coeff_Field = *averaged_coeff_field*)

Summary

Boundary condition for loose coupling of scalars to a free fluid region

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
mass_fraction_field	“string”	–
mass_fraction_diffusive_flux_field	“string”	–
averaged_coeff_field	“string”	–

Porous_Robin_Coupled_One_Region

Syntax

Bc Diffuse For Cvfem_Mass_Fraction ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Porous_Robin_Coupled_One_Region [Toggle = *toggle*]

Summary

Boundary condition for loose coupling of scalars to a free fluid region

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–

Scalar_String_Function

Syntax

Bc Diffuse For Cvfem_Mass_Fraction ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f	"string"	–

User_Field

Syntax

Bc Diffuse For Cvfem_Mass_Fraction ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc Diffuse For Cvfem_Mass_Fraction ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc Diffuse For Cvfem_Mass_Fraction ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

Cvfem_Mixture_Fraction

Constant

Syntax

Bc Diffuse For Cvfem_Mixture_Fraction ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | Value = *value*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
value	real	–

Exponential

Syntax

Bc Diffuse For Cvfem_Mixture_Fraction ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Free_Open_Flow

Syntax

Bc Diffuse For Cvfem_Mixture_Fraction ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Free_Open_Flow [Toggle = *toggle*]

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–

Global

Syntax

Bc Diffuse For Cvfem_Mixture_Fraction ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
global_name	“string”	–
component	integer	0

Mass_Flux

Syntax

Bc Diffuse For Cvfem_Mixture_Fraction ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Mass_Flux ([Toggle = *toggle*] | Total_Mdot = *total_mdot* | [Inflow_Phi = *inflow_phi*])

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
total_mdot	real	–
inflow_phi	real	0

Nd_Table

Syntax

Bc Diffuse For Cvfem_Mixture_Fraction ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Open_Adv_Flow

Syntax

Bc Diffuse For Cvfem_Mixture_Fraction ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Open_Adv_Flow ([Toggle = *toggle*] | [Pressure = *pressure*] | [Total_Pressure = *total_pressure*] | [Far_Field_Entrainment_Value = *far_field_entrainment_value*])

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
pressure	real	0
total_pressure	real	0
far_field_entrainment_value	real	0

Polynomial

Syntax

Bc Diffuse For Cvfem_Mixture_Fraction ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Diffuse For Cvfem_Mixture_Fraction ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f	"string"	–

User_Field

Syntax

Bc Diffuse For Cvfem_Mixture_Fraction ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc Diffuse For Cvfem_Mixture_Fraction ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc Diffuse For Cvfem_Mixture_Fraction ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

Cvfem_Momentum

Constant

Syntax

Bc Diffuse For Cvfem_Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | [Valuex = *valuex*] | [Valuey = *valuey*] | [Valuez = *valuez*])

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
valuex	real	0
valuey	real	0
valuez	real	0

Constant_Traction

Syntax

Bc Diffuse For Cvfem_Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant_Traction ([Toggle = *toggle*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
x	real	0
y	real	0
z	real	0

Exponential_Vapor_Recoil_Pressure

Syntax

Bc Diffuse For Cvfem_Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential_Vapor_Recoil_Pressure ([Toggle = *toggle*] | Tboil = *tboil* | L = *l* | Pa = *pa* | [Beta = *beta*] | Mw = *mw* | [Tl = *tl*] | [Th = *th*])

Summary

Exponential vapor recoil pressure momentum flux

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
tboil	real	–
l	real	–
pa	real	–
beta	real	1
mw	real	–
tl	real	-1
th	real	-1

Free_Open_Flow

Syntax

Bc Diffuse For Cvfem_Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Free_Open_Flow ([Toggle = *toggle*] | Pressure = *pressure*)

Summary

Open flow for momentum; apply the full stress

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
pressure	real	–

Interface

Syntax

Bc Diffuse For Cvfem_Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Interface [Toggle = *toggle*]

Summary

INTERFACE CVFEM momentum flux

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–

Mass_Flux

Syntax

Bc Diffuse For Cvfem_Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Mass_Flux ([Toggle = *toggle*] | Total_Mdot = *total_mdot* | [Temperature = *temperature*] | [Pressure = *pressure*] | [Mass_Fraction_0 = *mass_fraction_0*] | [Mass_Fraction_1 = *mass_fraction_1*] | [Mass_Fraction_2 = *mass_fraction_2*] | [Mass_Fraction_3 = *mass_fraction_3*] | [Mass_Fraction_4 = *mass_fraction_4*] | [Mass_Fraction_5 = *mass_fraction_5*] | [Mass_Fraction_6 = *mass_fraction_6*] | [Mass_Fraction_7 = *mass_fraction_7*] | [Mass_Fraction_8 = *mass_fraction_8*])

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
total_mdot	real	–
temperature	real	0
pressure	real	0
mass_fraction_0	real	0
mass_fraction_1	real	0
mass_fraction_2	real	0
mass_fraction_3	real	0
mass_fraction_4	real	0
mass_fraction_5	real	0
mass_fraction_6	real	0
mass_fraction_7	real	0
mass_fraction_8	real	0

Open_Adv_Flow

Syntax

Bc Diffuse For Cvfem_Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Open_Adv_Flow ([Toggle = *toggle*] | [Pressure = *pressure*] | [Total_Pressure = *total_pressure*] | [Coflow_Entrainment_Value_X = *coflow_entrainment_value_x*] | [Coflow_Entrainment_Value_Y = *coflow_entrainment_value_y*] | [Coflow_Entrainment_Value_Z = *coflow_entrainment_value_z*])

Summary

CVFEM momentum open advective flow flux BC

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is

active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
pressure	real	0
total_pressure	real	0
coflow_entrainment_value_x	real	0
coflow_entrainment_value_y	real	0
coflow_entrainment_value_z	real	0

Open_Flow

Syntax

Bc Diffuse For Cvfem_Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Open_Flow ([Toggle = *toggle*] | [Pressure = *pressure*] | [Total_Pressure = *total_pressure*])

Summary

Open flow for momentum; enforce normal gradient to be zero

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
pressure	real	0
total_pressure	real	0

Slip

Syntax

Bc Diffuse For Cvfem_Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Slip ([Toggle = *toggle*] | [Vs_X = *vs_x*] | [Vs_Y = *vs_y*] | [Vs_Z = *vs_z*] | Beta = *beta*)

Summary

Slip momentum flux

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
vs_x	real	0
vs_y	real	0
vs_z	real	0
beta	real	–

Slip_Length

Syntax

Bc Diffuse For Cvfem_Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Slip_Length ([Toggle = *toggle*] | [Vs_X = *vs_x*] | [Vs_Y = *vs_y*] | [Vs_Z = *vs_z*] | Beta = *beta*)

Summary

Slip length momentum flux

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
vs_x	real	0
vs_y	real	0
vs_z	real	0
beta	real	–

Symmetry_Flow

Syntax

Bc Diffuse For Cvfem_Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Symmetry_Flow [Toggle = *toggle*]

Summary

CVFEM momentum symmetry flow flux BC

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–

Vector_String_Function

Syntax

Bc Diffuse For Cvfem_Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Vector_String_Function ([Toggle = *toggle*] | F_X = *f_x* | F_Y = *f_y* | [F_Z = *f_z*])

Summary

Value from a string function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f_x	"string"	–
f_y	"string"	–
f_z	"string"	–

Wall_Function

Syntax

Bc Diffuse For Cvfem_Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Wall_Function ([Toggle = *toggle*] | [Wall_Velocity_X = *wall_velocity_x*] | [Wall_Velocity_Y = *wall_velocity_y*] | [Wall_Velocity_Z = *wall_velocity_z*] | Wall_Friction_Factor = *wall_friction_factor*)

Summary

CVFEM momentum wall function flux BC

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
wall_velocity_x	real	0
wall_velocity_y	real	0
wall_velocity_z	real	0
wall_friction_factor	real	–

Cvfem_Momentum_X

Constant

Syntax

Bc Diffuse For Cvfem_Momentum_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent*

Name ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | Value = *value*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
value	real	–

Exponential

Syntax

Bc Diffuse For Cvfem_Momentum_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Diffuse For Cvfem_Momentum_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Bc Diffuse For Cvfem_Momentum_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc Diffuse For Cvfem_Momentum_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Diffuse For Cvfem_Momentum_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f	"string"	–

User_Field

Syntax

Bc Diffuse For Cvfem_Momentum_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc Diffuse For Cvfem_Momentum_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc Diffuse For Cvfem_Momentum_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

Cvfem_Momentum_Y

Constant

Syntax

Bc Diffuse For Cvfem_Momentum_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | Value = *value*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
value	real	–

Exponential

Syntax

Bc Diffuse For Cvfem_Momentum_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Diffuse For Cvfem_Momentum_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Bc Diffuse For Cvfem_Momentum_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc Diffuse For Cvfem_Momentum_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Diffuse For Cvfem_Momentum_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f	"string"	–

User_Field

Syntax

Bc Diffuse For Cvfem_Momentum_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc Diffuse For Cvfem_Momentum_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc Diffuse For Cvfem_Momentum_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

Cvfem_Momentum_Z

Constant

Syntax

Bc Diffuse For Cvfem_Momentum_Z ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | Value = *value*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
value	real	–

Exponential

Syntax

Bc Diffuse For Cvfem_Momentum_Z ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Diffuse For Cvfem_Momentum_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Bc Diffuse For Cvfem_Momentum_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc Diffuse For Cvfem_Momentum_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Diffuse For Cvfem_Momentum_Z ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f	"string"	–

User_Field

Syntax

Bc Diffuse For Cvfem_Momentum_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc Diffuse For Cvfem_Momentum_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc Diffuse For Cvfem_Momentum_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

Cvfem_Pressure_Edge_Gradient_Projection

Constant

Syntax

Bc Diffuse For Cvfem_Pressure_Edge_Gradient_Projection ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | Value = *value*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
value	real	–

Exponential

Syntax

Bc Diffuse For Cvfem_Pressure_Edge_Gradient_Projection ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Diffuse For Cvfem_Pressure_Edge_Gradient_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Bc Diffuse For Cvfem_Pressure_Edge_Gradient_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc Diffuse For Cvfem_Pressure_Edge_Gradient_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Diffuse For Cvfem_Pressure_Edge_Gradient_Projection ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f	"string"	–

Simple_Interp

Syntax

Bc Diffuse For Cvfem_Pressure_Edge_Gradient_Projection ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Simple_Interp [Toggle = *toggle*]

Summary

CVFEM edge-based projection simple interpolation flux BC

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–

User_Field

Syntax

Bc Diffuse For Cvfem_Pressure_Edge_Gradient_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
scaling	real	1
global_var	“string”	–
time_function	“string”	–

User_Function

Syntax

Bc Diffuse For Cvfem_Pressure_Edge_Gradient_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc Diffuse For Cvfem_Pressure_Edge_Gradient_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

Cvfem_Projection

Constant

Syntax

Bc Diffuse For Cvfem_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | Value = *value*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
value	real	–

Exponential

Syntax

Bc Diffuse For Cvfem_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Diffuse For Cvfem_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Bc Diffuse For Cvfem_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Open

Syntax

Bc Diffuse For Cvfem_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Open ([Toggle = *toggle*] | [Total_Pressure = *total_pressure*] | [Pressure = *pressure*])

Summary

CVFEM projection open flux BC

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
total_pressure	real	0
pressure	real	0

Polynomial

Syntax

Bc Diffuse For Cvfem_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Diffuse For Cvfem_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f	"string"	–

Simple_Interp

Syntax

Bc Diffuse For Cvfem_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Simple_Interp [Toggle = *toggle*]

Summary

CVFEM projection simple interpolation flux BC

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–

User_Field

Syntax

Bc Diffuse For Cvfem_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
scaling	real	1
global_var	“string”	–
time_function	“string”	–

User_Function

Syntax

Bc Diffuse For Cvfem_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc Diffuse For Cvfem_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

Cvfem_Sdr_Edge_Gradient_Projection

Constant

Syntax

Bc Diffuse For Cvfem_Sdr_Edge_Gradient_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | Value = *value*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
value	real	–

Exponential

Syntax

Bc Diffuse For Cvfem_Sdr_Edge_Gradient_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Diffuse For Cvfem_Sdr_Edge_Gradient_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Bc Diffuse For Cvfem_Sdr_Edge_Gradient_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc Diffuse For Cvfem_Sdr_Edge_Gradient_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Diffuse For Cvfem_Sdr_Edge_Gradient_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f	"string"	–

Simple_Interp

Syntax

Bc Diffuse For Cvfem_Sdr_Edge_Gradient_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Simple_Interp [Toggle = *toggle*]

Summary

CVFEM edge-based projection simple interpolation flux BC

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–

User_Field

Syntax

Bc Diffuse For Cvfem_Sdr_Edge_Gradient_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
scaling	real	1
global_var	“string”	–
time_function	“string”	–

User_Function

Syntax

Bc Diffuse For Cvfem_Sdr_Edge_Gradient_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc Diffuse For Cvfem_Sdr_Edge_Gradient_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

Cvfem_Solvent_Continuity

Constant

Syntax

Bc Diffuse For Cvfem_Solvent_Continuity ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | Value = *value*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
value	real	–

Exponential

Syntax

Bc Diffuse For Cvfem_Solvent_Continuity ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Diffuse For Cvfem_Solvent_Continuity ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
global_name	"string"	–
component	integer	0

Inflow

Syntax

Bc Diffuse For Cvfem_Solvent_Continuity ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Inflow [Toggle = *toggle*]

Summary

Inflow mass flux for continuity equation; nodal interpolation

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–

Nd_Table

Syntax

Bc Diffuse For Cvfem_Solvent_Continuity ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Open_Flow

Syntax

Bc Diffuse For Cvfem_Solvent_Continuity ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Open_Flow ([Toggle = *toggle*] | [Pressure = *pressure*] | [Total_Pressure = *total_pressure*])

Summary

Open bc for continuity

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
pressure	real	0
total_pressure	real	0

Polynomial

Syntax

Bc Diffuse For Cvfem_Solvent_Continuity ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Diffuse For Cvfem_Solvent_Continuity ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f	"string"	–

Simple_Inflow

Syntax

Bc Diffuse For Cvfem_Solvent_Continuity ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Simple_Inflow ([Toggle = *toggle*] | *{density | value}* = *Density* | [Velocity_X = *Velocity_X*] | [Velocity_Y = *Velocity_Y*] | [Velocity_Z = *Velocity_Z*])

Summary

Inflow mass flux for continuity equation; specified density and velocity

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Density	real	–
Velocity_X	real	0
Velocity_Y	real	0
Velocity_Z	real	0

User_Field

Syntax

Bc Diffuse For Cvfem_Solvent_Continuity ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc Diffuse For Cvfem_Solvent_Continuity ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc Diffuse For Cvfem_Solvent_Continuity ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

Cvfem_Solvent_Momentum

Constant

Syntax

Bc Diffuse For Cvfem_Solvent_Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | [Valuex = *valuex*] | [Valuey = *valuey*] | [Valuez = *valuez*])

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
valuex	real	0
valuey	real	0
valuez	real	0

Open_Adv_Flow

Syntax

Bc Diffuse For Cvfem_Solvent_Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Open_Adv_Flow ([Toggle = *toggle*] | [Pressure = *pressure*] | [Total_Pressure = *total_pressure*] | [Coflow_Entrainment_Value_X = *coflow_entrainment_value_x*] | [Coflow_Entrainment_Value_Y = *coflow_entrainment_value_y*] | [Coflow_Entrainment_Value_Z = *coflow_entrainment_value_z*])

Summary

CVFEM solvent momentum open advective flow flux BC

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
pressure	real	0
total_pressure	real	0
coflow_entrainment_value_x	real	0
coflow_entrainment_value_y	real	0
coflow_entrainment_value_z	real	0

Open_Flow

Syntax

Bc Diffuse For Cvfem_Solvent_Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Open_Flow ([Toggle = *toggle*] | [Pressure = *pressure*] | [Total_Pressure = *total_pressure*])

Summary

CVFEM solvent momentum open flow flux BC

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
pressure	real	0
total_pressure	real	0

Symmetry_Flow

Syntax

Bc Diffuse For Cvfem_Solvent_Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Symmetry_Flow [Toggle = *toggle*]

Summary

CVFEM solvent momentum symmetry flow flux BC

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–

Vector_String_Function

Syntax

Bc Diffuse For Cvfem_Solvent_Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Vector_String_Function ([Toggle = *toggle*] | F_X = *f_x* | F_Y = *f_y* | [F_Z = *f_z*])

Summary

Value from a string function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
f_x	“string”	–
f_y	“string”	–
f_z	“string”	–

Cvfem_Solvent_Momentum_X

Constant

Syntax

Bc Diffuse For Cvfem_Solvent_Momentum_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | Value = *value*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
value	real	–

Exponential

Syntax

Bc Diffuse For Cvfem_Solvent_Momentum_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Diffuse For Cvfem_Solvent_Momentum_X ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Bc Diffuse For Cvfem_Solvent_Momentum_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc Diffuse For Cvfem_Solvent_Momentum_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Diffuse For Cvfem_Solvent_Momentum_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f	"string"	–

User_Field

Syntax

Bc Diffuse For Cvfem_Solvent_Momentum_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc Diffuse For Cvfem_Solvent_Momentum_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc Diffuse For Cvfem_Solvent_Momentum_X ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

Cvfem_Solvent_Momentum_Y

Constant

Syntax

Bc Diffuse For Cvfem_Solvent_Momentum_Y ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | Value = *value*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
value	real	–

Exponential

Syntax

Bc Diffuse For Cvfem_Solvent_Momentum_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Diffuse For Cvfem_Solvent_Momentum_Y ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Bc Diffuse For Cvfem_Solvent_Momentum_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc Diffuse For Cvfem_Solvent_Momentum_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Diffuse For Cvfem_Solvent_Momentum_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f	"string"	–

User_Field

Syntax

Bc Diffuse For Cvfem_Solvent_Momentum_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc Diffuse For Cvfem_Solvent_Momentum_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc Diffuse For Cvfem_Solvent_Momentum_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

Cvfem_Solvent_Momentum_Z

Constant

Syntax

Bc Diffuse For Cvfem_Solvent_Momentum_Z ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | Value = *value*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
value	real	–

Exponential

Syntax

Bc Diffuse For Cvfem_Solvent_Momentum_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Diffuse For Cvfem_Solvent_Momentum_Z ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Bc Diffuse For Cvfem_Solvent_Momentum_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc Diffuse For Cvfem_Solvent_Momentum_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Diffuse For Cvfem_Solvent_Momentum_Z ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f	"string"	–

User_Field

Syntax

Bc Diffuse For Cvfem_Solvent_Momentum_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc Diffuse For Cvfem_Solvent_Momentum_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc Diffuse For Cvfem_Solvent_Momentum_Z ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

Cvfem_Specific_Dissipation_Rate

Constant

Syntax

Bc Diffuse For Cvfem_Specific_Dissipation_Rate ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | Value = *value*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
value	real	–

Exponential

Syntax

Bc Diffuse For Cvfem_Specific_Dissipation_Rate ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Diffuse For Cvfem_Specific_Dissipation_Rate ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
global_name	"string"	–
component	integer	0

Mass_Flux

Syntax

Bc Diffuse For Cvfem_Specific_Dissipation_Rate ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Mass_Flux ([Toggle = *toggle*] | Total_Mdot = *total_mdot* | [Inflow_Phi = *inflow_phi*])

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
total_mdot	real	–
inflow_phi	real	0

Nd_Table

Syntax

Bc Diffuse For Cvfem_Specific_Dissipation_Rate ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Open_Adv_Flow

Syntax

Bc Diffuse For Cvfem_Specific_Dissipation_Rate ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Open_Adv_Flow ([Toggle = *toggle*] | [Pressure = *pressure*] | [Total_Pressure = *total_pressure*] | [Far_Field_Entrainment_Value = *far_field_entrainment_value*])

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
pressure	real	0
total_pressure	real	0
far_field_entrainment_value	real	0

Polynomial

Syntax

Bc Diffuse For Cvfem_Specific_Dissipation_Rate ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
variable	“string”	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Diffuse For Cvfem_Specific_Dissipation_Rate ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f	"string"	–

User_Field

Syntax

Bc Diffuse For Cvfem_Specific_Dissipation_Rate ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc Diffuse For Cvfem_Specific_Dissipation_Rate ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc Diffuse For Cvfem_Specific_Dissipation_Rate ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

Cvfem_Temperature_Edge_Gradient_Projection

Constant

Syntax

Bc Diffuse For Cvfem_Temperature_Edge_Gradient_Projection ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | Value = *value*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
value	real	–

Exponential

Syntax

Bc Diffuse For Cvfem_Temperature_Edge_Gradient_Projection ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Diffuse For Cvfem_Temperature_Edge_Gradient_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Bc Diffuse For Cvfem_Temperature_Edge_Gradient_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc Diffuse For Cvfem_Temperature_Edge_Gradient_Projection ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Diffuse For Cvfem_Temperature_Edge_Gradient_Projection ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f	"string"	–

Simple_Interp

Syntax

Bc Diffuse For Cvfem_Temperature_Edge_Gradient_Projection ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Simple_Interp [Toggle = *toggle*]

Summary

CVFEM edge-based projection simple interpolation flux BC

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–

User_Field

Syntax

Bc Diffuse For Cvfem_Temperature_Edge_Gradient_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
scaling	real	1
global_var	“string”	–
time_function	“string”	–

User_Function

Syntax

Bc Diffuse For Cvfem_Temperature_Edge_Gradient_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc Diffuse For Cvfem_Temperature_Edge_Gradient_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
Name	“string”	–
plugin_parameters	“string”	–

Cvfem_Tke_Edge_Gradient_Projection

Constant

Syntax

Bc Diffuse For Cvfem_Tke_Edge_Gradient_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | Value = *value*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
value	real	–

Exponential

Syntax

Bc Diffuse For Cvfem_Tke_Edge_Gradient_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Diffuse For Cvfem_Tke_Edge_Gradient_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Bc Diffuse For Cvfem_Tke_Edge_Gradient_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc Diffuse For Cvfem_Tke_Edge_Gradient_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Diffuse For Cvfem_Tke_Edge_Gradient_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f	"string"	–

Simple_Interp

Syntax

Bc Diffuse For Cvfem_Tke_Edge_Gradient_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Simple_Interp [Toggle = *toggle*]

Summary

CVFEM edge-based projection simple interpolation flux BC

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–

User_Field

Syntax

Bc Diffuse For Cvfem_Tke_Edge_Gradient_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
scaling	real	1
global_var	“string”	–
time_function	“string”	–

User_Function

Syntax

Bc Diffuse For Cvfem_Tke_Edge_Gradient_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc Diffuse For Cvfem_Tke_Edge_Gradient_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

Cvfem_Turbulence_Dissipation_Rate

Constant

Syntax

Bc Diffuse For Cvfem_Turbulence_Dissipation_Rate ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | Value = *value*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
value	real	–

Exponential

Syntax

Bc Diffuse For Cvfem_Turbulence_Dissipation_Rate ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Free_Open_Flow

Syntax

Bc Diffuse For Cvfem_Turbulence_Dissipation_Rate ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Free_Open_Flow [Toggle = *toggle*]

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–

Global

Syntax

Bc Diffuse For Cvfem_Turbulence_Dissipation_Rate ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
global_name	“string”	–
component	integer	0

Mass_Flux

Syntax

Bc Diffuse For Cvfem_Turbulence_Dissipation_Rate ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Mass_Flux ([Toggle = *toggle*] | Total_Mdot = *total_mdot* | [Inflow_Phi = *inflow_phi*])

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
total_mdot	real	–
inflow_phi	real	0

Nd_Table

Syntax

Bc Diffuse For Cvfem_Turbulence_Dissipation_Rate ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Open_Adv_Flow

Syntax

Bc Diffuse For Cvfem_Turbulence_Dissipation_Rate ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Open_Adv_Flow ([Toggle = *toggle*] | [Pressure = *pressure*] | [Total_Pressure = *total_pressure*] | [Far_Field_Entrainment_Value = *far_field_entrainment_value*])

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
pressure	real	0
total_pressure	real	0
far_field_entrainment_value	real	0

Polynomial

Syntax

Bc Diffuse For Cvfem_Turbulence_Dissipation_Rate ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Diffuse For Cvfem_Turbulence_Dissipation_Rate ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f	"string"	–

User_Field

Syntax

Bc Diffuse For Cvfem_Turbulence_Dissipation_Rate ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc Diffuse For Cvfem_Turbulence_Dissipation_Rate ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc Diffuse For Cvfem_Turbulence_Dissipation_Rate ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

Cvfem_Turbulent_Kinetic_Energy

Constant

Syntax

Bc Diffuse For Cvfem_Turbulent_Kinetic_Energy ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | Value = *value*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
value	real	–

Exponential

Syntax

Bc Diffuse For Cvfem_Turbulent_Kinetic_Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Free_Open_Flow

Syntax

Bc Diffuse For Cvfem_Turbulent_Kinetic_Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Free_Open_Flow [Toggle = *toggle*]

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–

Global

Syntax

Bc Diffuse For Cvfem_Turbulent_Kinetic_Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
global_name	“string”	–
component	integer	0

Mass_Flux

Syntax

Bc Diffuse For Cvfem_Turbulent_Kinetic_Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Mass_Flux ([Toggle = *toggle*] | Total_Mdot = *total_mdot* | [Inflow_Phi = *inflow_phi*])

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
total_mdot	real	–
inflow_phi	real	0

Nd_Table

Syntax

Bc Diffuse For Cvfem_Turbulent_Kinetic_Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Open_Adv_Flow

Syntax

Bc Diffuse For Cvfem_Turbulent_Kinetic_Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Open_Adv_Flow ([Toggle = *toggle*] | [Pressure = *pressure*] | [Total_Pressure = *total_pressure*] | [Far_Field_Entrainment_Value = *far_field_entrainment_value*])

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
pressure	real	0
total_pressure	real	0
far_field_entrainment_value	real	0

Polynomial

Syntax

Bc Diffuse For Cvfem_Turbulent_Kinetic_Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Diffuse For Cvfem_Turbulent_Kinetic_Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f	"string"	–

User_Field

Syntax

Bc Diffuse For Cvfem_Turbulent_Kinetic_Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc Diffuse For Cvfem_Turbulent_Kinetic_Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc Diffuse For Cvfem_Turbulent_Kinetic_Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

Cvfem_Velocity_Edge_Gradient_Projection

Constant

Syntax

Bc Diffuse For Cvfem_Velocity_Edge_Gradient_Projection ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | Value = *value*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
value	real	–

Exponential

Syntax

Bc Diffuse For Cvfem_Velocity_Edge_Gradient_Projection ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Diffuse For Cvfem_Velocity_Edge_Gradient_Projection ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Bc Diffuse For Cvfem_Velocity_Edge_Gradient_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc Diffuse For Cvfem_Velocity_Edge_Gradient_Projection ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Diffuse For Cvfem_Velocity_Edge_Gradient_Projection ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f	"string"	–

User_Field

Syntax

Bc Diffuse For Cvfem_Velocity_Edge_Gradient_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc Diffuse For Cvfem_Velocity_Edge_Gradient_Projection ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc Diffuse For Cvfem_Velocity_Edge_Gradient_Projection ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

Cvfem_Velocity_Pressure_Projection

Constant

Syntax

Bc Diffuse For Cvfem_Velocity_Pressure_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | [Valuex = *valuex*] | [Valuey = *valuey*] | [Valuez = *valuez*])

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
valuex	real	0
valuey	real	0
valuez	real	0

Simple_Dp_Interp

Syntax

Bc Diffuse For Cvfem_Velocity_Pressure_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Simple_Dp_Interp [Toggle = *toggle*]

Summary

Simple DP interpolation model for CVFEM velocity pressure projection flux

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–

Vector_String_Function

Syntax

Bc Diffuse For Cvfem_Velocity_Pressure_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Vector_String_Function ([Toggle = *toggle*] | F_X = *f_x* | F_Y = *f_y* | [F_Z = *f_z*])

Summary

Value from a string function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
f_x	“string”	–
f_y	“string”	–
f_z	“string”	–

Cvfem_Velocity_Pressure_Projection_X

Constant

Syntax

Bc Diffuse For Cvfem_Velocity_Pressure_Projection_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | Value = *value*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
value	real	–

Exponential

Syntax

Bc Diffuse For Cvfem_Velocity_Pressure_Projection_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Diffuse For Cvfem_Velocity_Pressure_Projection_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Bc Diffuse For Cvfem_Velocity_Pressure_Projection_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc Diffuse For Cvfem_Velocity_Pressure_Projection_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Diffuse For Cvfem_Velocity_Pressure_Projection_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f	"string"	–

User_Field

Syntax

Bc Diffuse For Cvfem_Velocity_Pressure_Projection_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc Diffuse For Cvfem_Velocity_Pressure_Projection_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc Diffuse For Cvfem_Velocity_Pressure_Projection_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

Cvfem_Velocity_Pressure_Projection_Y

Constant

Syntax

Bc Diffuse For Cvfem_Velocity_Pressure_Projection_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | Value = *value*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
value	real	–

Exponential

Syntax

Bc Diffuse For Cvfem_Velocity_Pressure_Projection_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Diffuse For Cvfem_Velocity_Pressure_Projection_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Bc Diffuse For Cvfem_Velocity_Pressure_Projection_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc Diffuse For Cvfem_Velocity_Pressure_Projection_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Diffuse For Cvfem_Velocity_Pressure_Projection_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f	"string"	–

User_Field

Syntax

Bc Diffuse For Cvfem_Velocity_Pressure_Projection_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc Diffuse For Cvfem_Velocity_Pressure_Projection_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc Diffuse For Cvfem_Velocity_Pressure_Projection_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
Name	“string”	–
plugin_parameters	“string”	–

Cvfem_Velocity_Pressure_Projection_Z

Constant

Syntax

Bc Diffuse For Cvfem_Velocity_Pressure_Projection_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | Value = *value*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
value	real	–

Exponential

Syntax

Bc Diffuse For Cvfem_Velocity_Pressure_Projection_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Diffuse For Cvfem_Velocity_Pressure_Projection_Z ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Bc Diffuse For Cvfem_Velocity_Pressure_Projection_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc Diffuse For Cvfem_Velocity_Pressure_Projection_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Diffuse For Cvfem_Velocity_Pressure_Projection_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f	"string"	–

User_Field

Syntax

Bc Diffuse For Cvfem_Velocity_Pressure_Projection_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc Diffuse For Cvfem_Velocity_Pressure_Projection_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc Diffuse For Cvfem_Velocity_Pressure_Projection_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

Darcy_Momentum

Constant

Syntax

Bc Diffuse For Darcy_Momentum ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | [Valuex = *valuex*] | [Valuey = *valuey*] | [Valuez = *valuez*])

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
valuex	real	0
valuey	real	0
valuez	real	0

Open_Flow

Syntax

Bc Diffuse For Darcy_Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Open_Flow ([Toggle = *toggle*] | [Pressure = *pressure*] | [Total_Pressure = *total_pressure*])

Summary

Open flow momentum flux

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
pressure	real	0
total_pressure	real	0

Pressure

Syntax

Bc Diffuse For Darcy_Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Pressure ([Toggle = *toggle*] | [User_Field_Name = *user_field_name*] | [Use_Bulk_Node = *use_bulk_node*] | [P = *p*] | [C_T = *c_t*] | [C_X = *c_x*] | [C_Y = *c_y*] | [C_Z = *c_z*] | [C_Cos_W = *c_cos_w*] | [C_Sin_W = *c_sin_w*] | [C1_Cos = *c1_cos*] | [C1_Sin = *c1_sin*])

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
user_field_name	"string"	–
use_bulk_node	"string"	–
p	real	0
c_t	real	0
c_x	real	0
c_y	real	0
c_z	real	0
c_cos_w	real	0
c_sin_w	real	0
c1_cos	real	0
c1_sin	real	0

Rt_Pressure

Syntax

Bc Diffuse For Darcy_Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Rt_Pressure ([Toggle = *toggle*] | P = *p* | [C_T = *c_t*] | [C_X = *c_x*] | [C_Y = *c_y*] | [C_Z = *c_z*])

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
p	real	–
c_t	real	0
c_x	real	0
c_y	real	0
c_z	real	0

Simple_Interp

Syntax

Bc Diffuse For Darcy_Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Simple_Interp [Toggle = *toggle*]

Summary

Darcy momentum simple interpolation flux BC

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–

Slip

Syntax

Bc Diffuse For Darcy_Momentum ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Slip ([Toggle = *toggle*] | [Vs_X = *vs_x*] | [Vs_Y = *vs_y*] | [Vs_Z = *vs_z*] | K = *k*)

Summary

Darcy slip momentum flux

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
vs_x	real	0
vs_y	real	0
vs_z	real	0
k	real	–

Vector_String_Function

Syntax

Bc Diffuse For Darcy_Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Vector_String_Function ([Toggle = *toggle*] | F_X = *f_x* | F_Y = *f_y* | [F_Z = *f_z*])

Summary

Value from a string function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
f_x	“string”	–
f_y	“string”	–
f_z	“string”	–

Darcy_Momentum_X

Constant

Syntax

Bc Diffuse For Darcy_Momentum_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | Value = *value*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
value	real	–

Exponential

Syntax

Bc Diffuse For Darcy_Momentum_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Diffuse For Darcy_Momentum_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
global_name	“string”	–
component	integer	0

Nd_Table

Syntax

Bc Diffuse For Darcy_Momentum_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc Diffuse For Darcy_Momentum_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Diffuse For Darcy_Momentum_X ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f	"string"	–

User_Field

Syntax

Bc Diffuse For Darcy_Momentum_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
scaling	real	1
global_var	“string”	–
time_function	“string”	–

User_Function

Syntax

Bc Diffuse For Darcy_Momentum_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc Diffuse For Darcy_Momentum_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

Darcy_Momentum_Y

Constant

Syntax

Bc Diffuse For Darcy_Momentum_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | Value = *value*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
value	real	–

Exponential

Syntax

Bc Diffuse For Darcy_Momentum_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Diffuse For Darcy_Momentum_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Bc Diffuse For Darcy_Momentum_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc Diffuse For Darcy_Momentum_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Diffuse For Darcy_Momentum_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f	"string"	–

User_Field

Syntax

Bc Diffuse For Darcy_Momentum_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc Diffuse For Darcy_Momentum_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc Diffuse For Darcy_Momentum_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

Darcy_Momentum_Z

Constant

Syntax

Bc Diffuse For Darcy_Momentum_Z ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | Value = *value*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
value	real	–

Exponential

Syntax

Bc Diffuse For Darcy_Momentum_Z ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Diffuse For Darcy_Momentum_Z ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Bc Diffuse For Darcy_Momentum_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc Diffuse For Darcy_Momentum_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Diffuse For Darcy_Momentum_Z ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f	"string"	–

User_Field

Syntax

Bc Diffuse For Darcy_Momentum_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc Diffuse For Darcy_Momentum_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc Diffuse For Darcy_Momentum_Z ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

Div_Projection

Constant

Syntax

Bc Diffuse For Div_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | [Valuex = *valuex*] | [Valuey = *valuey*] | [Valuez = *valuez*])

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
valuex	real	0
valuey	real	0
valuez	real	0

Vector_String_Function

Syntax

Bc Diffuse For Div_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Vector_String_Function ([Toggle = *toggle*] | F_X = *f_x* | F_Y = *f_y* | [F_Z = *f_z*])

Summary

Value from a string function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f_x	"string"	–
f_y	"string"	–
f_z	"string"	–

Div_Projection_X

Constant

Syntax

Bc Diffuse For Div_Projection_X ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | Value = *value*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
value	real	–

Exponential

Syntax

Bc Diffuse For Div_Projection_X ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Diffuse For Div_Projection_X ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Bc Diffuse For Div_Projection_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc Diffuse For Div_Projection_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Diffuse For Div_Projection_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f	"string"	–

User_Field

Syntax

Bc Diffuse For Div_Projection_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc Diffuse For Div_Projection_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc Diffuse For Div_Projection_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

Div_Projection_Y

Constant

Syntax

Bc Diffuse For Div_Projection_Y ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | Value = *value*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
value	real	–

Exponential

Syntax

Bc Diffuse For Div_Projection_Y ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Diffuse For Div_Projection_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Bc Diffuse For Div_Projection_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc Diffuse For Div_Projection_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Diffuse For Div_Projection_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f	"string"	–

User_Field

Syntax

Bc Diffuse For Div_Projection_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
scaling	real	1
global_var	“string”	–
time_function	“string”	–

User_Function

Syntax

Bc Diffuse For Div_Projection_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc Diffuse For Div_Projection_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

Div_Projection_Z

Constant

Syntax

Bc Diffuse For Div_Projection_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | Value = *value*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
value	real	–

Exponential

Syntax

Bc Diffuse For Div_Projection_Z ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
variable	“string”	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Diffuse For Div_Projection_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
global_name	“string”	–
component	integer	0

Nd_Table

Syntax

Bc Diffuse For Div_Projection_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc Diffuse For Div_Projection_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Diffuse For Div_Projection_Z ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f	"string"	–

User_Field

Syntax

Bc Diffuse For Div_Projection_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
scaling	real	1
global_var	“string”	–
time_function	“string”	–

User_Function

Syntax

Bc Diffuse For Div_Projection_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc Diffuse For Div_Projection_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
Name	“string”	–
plugin_parameters	“string”	–

Energy

Calore_User_Sub

Syntax

Bc Diffuse For Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Calore_User_Sub ([Power_Output = *power_output*] | [Flux_Output = *flux_output*] | [Toggle = *toggle*] | Name = *name* | Type = *type* | [Multiplier = *multiplier*] | [Material_Data_Block = *material_data_block*] | [Data = *data*] | [Scaling_Field = *scaling_field*])

Summary

Values from a Calore user subroutine

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output

If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Flux_Output

If specified, the average flux of the given BC/Src is calculated and stored into the global variable name specified as the *flux_output* argument.

Toggle

If given, specifies which *toggle block* controls whether this BC is

active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
power_output	"string"	–
flux_output	"string"	–
toggle	"string"	–
name	"string"	–
type	"string"	–
multiplier	real	1
material_data_block	"string"	–
data	"string"	–
scaling_field	"string"	–

Cht_Flux

Syntax

Bc Diffuse For Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Cht_Flux ([Power_Output = *power_output*] | [Flux_Output = *flux_output*] | [Toggle = *toggle*] | [Too_Field = *Too_Field*] | [H_Field = *H_Field*])

Summary

Applies H(T-Too) flux for conjugate heat transfer boundary. Optional parameters are "Too_Field" and "H_Field", which default to "nodal_cht_Too" and "nodal_cht_H" respectively.

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output

If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Flux_Output

If specified, the average flux of the given BC/Src is calculated and stored into the global variable name specified as the *flux_output* argument.

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
power_output	"string"	–
flux_output	"string"	–
toggle	"string"	–
Too_Field	"string"	nodal_cht_Too
H_Field	"string"	nodal_cht_H

Cht_Robin

Syntax

Bc Diffuse For Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Cht_Robin ([Power_Output = *power_output*] | [Flux_Output = *flux_output*] | [Toggle = *toggle*])

Summary

Applies a Robin-style BC of the form: J dot n + h * (T-Tref).

Description

J, h, and Tref should be specified using the material properties BC_HEAT_FLUX, HEAT_TRANSFER_COEFFICIENT, and BC_REFERENCE_TEMPERATURE in a material that is applied to the surface.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output

If specified, the total (integrated) power of the given BC/Src is

calculated and stored into the global variable name specified as the *power_output* argument.

Flux_Output

If specified, the average flux of the given BC/Src is calculated and stored into the global variable name specified as the *flux_output* argument.

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
power_output	"string"	–
flux_output	"string"	–
toggle	"string"	–

Constant

Syntax

Bc Diffuse For Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Power_Output = *power_output*] | [Flux_Output = *flux_output*] | [Toggle = *toggle*] | *{flux | value}* = *flux*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output

If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Flux_Output

If specified, the average flux of the given BC/Src is calculated and stored into the global variable name specified as the *flux_output* argument.

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
power_output	"string"	–
flux_output	"string"	–
toggle	"string"	–
flux	real	–

Distribution_Factor

Syntax

Bc Diffuse For Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Distribution_Factor ([Power_Output = *power_output*] | [Flux_Output = *flux_output*] | [Toggle = *toggle*] | [Attribute_Name = *attribute_name*] | [Multiplier = *multiplier*])

Summary

Applies a flux equal to a distribution factor defined in the input mesh with name ATTRIBUTE_NAME. F = MULTIPLIER x ATTRIBUTE_NAME[node]

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output

If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Flux_Output

If specified, the average flux of the given BC/Src is calculated and stored into the global variable name specified as the *flux_output* argument.

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
power_output	"string"	–
flux_output	"string"	–
toggle	"string"	–
attribute_name	"string"	DISTRIBUTION_FACTOR
multiplier	real	1

Enclosure_Radiation

Syntax

Bc Diffuse For Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Enclosure_Radiation ([Power_Output = *power_output*] | [Flux_Output = *flux_output*] | [Toggle = *toggle*] | [Multiplier = *multiplier*] | Enclosure = *enclosure*)

Summary

This term imposes an energy flux due to enclosure radiation as defined in *Enclosure Radiation*. The irradiation is provided by the Chapparal radiosity solution

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output

If specified, the total (integrated) power of the given BC/Src is

calculated and stored into the global variable name specified as the *power_output* argument.

Flux_Output

If specified, the average flux of the given BC/Src is calculated and stored into the global variable name specified as the *flux_output* argument.

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
power_output	"string"	–
flux_output	"string"	–
toggle	"string"	–
multiplier	real	1
enclosure	"string"	–

Exponential

Syntax

Bc Diffuse For Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Power_Output = *power_output*] | [Flux_Output = *flux_output*] | [Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

$$F = C + Me^{kX}$$

where X is the variable, C is the constant, M is the multiplier, and k is exponent. C and M are optional and default to 0 and 1 respectively.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output
If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Flux_Output
If specified, the average flux of the given BC/Src is calculated and stored into the global variable name specified as the *flux_output* argument.

Toggle
If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
power_output	"string"	–
flux_output	"string"	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Exponential_Vapor_Cooling

Syntax

Bc Diffuse For Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential_Vapor_Cooling ([Power_Output = *power_output*] | [Flux_Output = *flux_output*] | [Toggle = *toggle*] | Tboil = *tboil* | L = *l* | Pa = *pa* | Mw = *mw* | Alpha = *alpha* | [Tl = *tl*] | [Th = *th*])

Summary

Exponential vapor cooling energy flux

Description

Flux is given by:

$$F = P_a L\alpha \exp\left(\frac{L}{R}\left(\frac{1}{T_{boil}} - \frac{1}{T}\right)\right)\sqrt{\frac{1}{2\pi RT}}$$

where R is the specific gas constant ($\bar{R}$/MW). For T less than T_L, zero flux is applied. For T between T_L and T_H, a 3rd order smoothing polynomial is used to provide continuity of values and derivatives between 0 and the expression given above. For T greater than T_H, the above expression is used. If not provided, T_L and T_H are set to 10% below and above T_{Boil}, respectively.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output

If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Flux_Output

If specified, the average flux of the given BC/Src is calculated and stored into the global variable name specified as the *flux_output* argument.

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
power_output	"string"	–
flux_output	"string"	–
toggle	"string"	–
tboil	real	–
l	real	–
pa	real	–
mw	real	–
alpha	real	–
tl	real	-1
th	real	-1

Face_Field_Scalar

Syntax

Bc Diffuse For Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Face_Field_Scalar ([Power_Output = *power_output*] | [Flux_Output = *flux_output*] | [Toggle = *toggle*] | Name = *name* | [Multiplier = *multiplier*] | [Advective_Bar = *advective_bar*] | [Command_Block_Name = *command_block_name*])

Summary

FACE or EDGE Field energy flux

Description

Flux is given by:

$$M \cdot f$$

M is given by MULTIPLIER and f is a face or edge field with name NAME.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output

If specified, the total (integrated) power of the given BC/Src is

calculated and stored into the global variable name specified as the *power_output* argument.

Flux_Output

If specified, the average flux of the given BC/Src is calculated and stored into the global variable name specified as the *flux_output* argument.

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
power_output	"string"	–
flux_output	"string"	–
toggle	"string"	–
name	"string"	–
multiplier	real	1
advective_bar	"string"	–
command_block_name	"string"	–

Fortran

Syntax

Bc Diffuse For Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Fortran ([Power_Output = *power_output*] | [Flux_Output = *flux_output*] | [Toggle = *toggle*] | [Multiplier = *multiplier*] | Sub_Name = *sub_name* | [Real_Data = *real_data*] | [Int_Data = *int_data*] | [Resource_Name = *resource_name*] | [Data = *data*])

Summary

Fortran subroutine energy flux

Description

The following options are present in most boundary conditions/sources. The

meaning of each option is as follows:

Power_Output
If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Flux_Output
If specified, the average flux of the given BC/Src is calculated and stored into the global variable name specified as the *flux_output* argument.

Toggle
If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
power_output	"string"	–
flux_output	"string"	–
toggle	"string"	–
multiplier	real	1
sub_name	"string"	–
real_data	"string"	–
int_data	"string"	–
resource_name	"string"	–
data	"string"	–

Gaussian_Power_Weld

Syntax

Bc Diffuse For Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Gaussian_Power_Weld ([Power_Output = *power_output*] | [Flux_Output = *flux_output*] | [Toggle = *toggle*] | R = *r* | [Clip_Radius = *clip_radius*] | [Efficiency = *efficiency*] | R_Eff = *R_eff* | Origin_And_Normal_Function =

origin_and_normal_function | [Old_Temperature = *old_temperature*] | Path_Function = *path_function* | [Deposition = *deposition*] | [Integrated = *integrated*] | [Clip = *clip*])

Summary

Energy flux due to an incident Gaussian laser beam

Description

The coordinates of the center of the beam are provided by the X, Y, and Z components of the provided origin_and_normal_function. The direction of the beam is provided by the NX, NY, and NZ components of the origin_and_normal_function. The applied flux is given by:

$$\frac{(-\boldsymbol{n})\cdot\boldsymbol{d}}{|\boldsymbol{d}|}\frac{\alpha e R_{eff} P}{\pi R^2(1-e^{-R_{eff}})}e^{-R_{eff}\frac{r_p^2}{R^2}}$$

P is the laser power defined for the material, R is the beam radius, R_{eff} is the effective beam radius, α is the laser absorptivity defined for the material, e is the EFFICIENCY, and r_p is the shortest distance between a point on the surface and the line defined by the beam origin and normal. The term $-\boldsymbol{n}\cdot\boldsymbol{d}$ accounts for the fact that the surface may not be orthogonal to the laser. Points with values of r_p > CLIP_RADIUS will receive zero flux. If ACTIVATION is set to true, points whose temperature exceed ACTIVATION_TEMPERATURE will be flagged as activated for use with the ACTIVATION USER FUNCTION thermal conductivity model. If INTEGRATED is true, this flux will be integrated between the previous time step and the current one and the average flux applied.

See the *Directed Energy* section for the appropriate syntax to use this flux in the directed energy block.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output

If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Flux_Output

If specified, the average flux of the given BC/Src is calculated and stored into the global variable name specified as the *flux_output* argument.

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
power_output	"string"	–
flux_output	"string"	–
toggle	"string"	–
r	real	–
clip_radius	real	0
efficiency	real	1
R_eff	real	–
origin_and_normal_function	"string"	–
old_temperature	"string"	TRUE
path_function	"string"	–
deposition	"string"	FALSE
integrated	"string"	FALSE
clip	"string"	TRUE

Gaussian_Ray_Trace

Syntax

Bc Diffuse For Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Gaussian_Ray_Trace ([Power_Output = *power_output*] | [Flux_Output = *flux_output*] | [Toggle = *toggle*] | Path_Function = *path_function* | Dir = *dir* | Power = *power* | Type = *type* | [N_Rays = *n_rays*] | [N_R = *n_r*] | [N_Theta_Initial = *n_theta_initial*] | [N_Intersections = *n_intersections*] | [Weight_Ratio = *weight_ratio*] | [Id = *id*] | [Absorption_Model = *absorption_model*] | [Parallel_Type = *parallel_type*] | R = *r* | R_Eff = *r_eff*)

Summary

Ray tracing implementation of a Gaussian Beam energy flux

Description

r - Laser Gaussian standard deviation.

r_eff - Laser exponential factor. Gaussian is given by $EXP\left(\frac{-r_eff*D^2}{r^2}\right)$, where D

is the distance between a point and the laser center.

type - Either radial discretization or Monte Carlo. Determines how beam will be discretized into rays. Radial discretization is deterministic and breaks beams into approximately equal area segments radially. Each segment generates one ray. The power of each ray varies based on which segment it is generated from. Monte Carlo is stochastic and will generate a user-specified number of rays distributed spatially according to the laser's shape function, with each ray having equal power of power/n_rays.

n_r - Number of segments in radial direction for radial discretization.

n_theta_initial - Number of segments in theta direction in the smallest radial segment. The number of theta segments will grow as radial direction increases so segments have approximately the same area. Total number of rays will be n_r*n_r*n_theta_initial.

n_rays - Number of rays to generate for Monte Carlo discretization.

path_function - Function giving x, y, z coordinates of laser center as function of time.

dir - Function giving x,y,z components of direction that rays will travel initially from laser center defined by path_function.

power - Function giving total laser power.

n_intersections - Maximum number of times to intersect a ray with mesh elements due to reflections (default: 10).

weight_ratio - Fraction of initial ray power below which rays will be considered spent and no longer be traced (default: 0.001).

absorption_model - Either Specular, to use a surface emissivity value to determine fraction of energy absorbed and reflected with each ray intersection or Fresnel to use Fresnel model dependent on ray angle of incidence.

parallel_type - Either ray_parallel, to divide rays between processors with a globally shared mesh part or scene_parallel, to divide mesh part between processors with a globally shared set of rays.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output
: If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Flux_Output
: If specified, the average flux of the given BC/Src is calculated and

stored into the global variable name specified as the *flux_output* argument.

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
power_output	“string”	–
flux_output	“string”	–
toggle	“string”	–
path_function	“string”	–
dir	“string”	–
power	“string”	–
type	“string”	–
n_rays	integer	0
n_r	integer	0
n_theta_initial	integer	0
n_intersections	integer	10
weight_ratio	real	0.001
id	“string”	–
absorption_model	“string”	SPECULAR
parallel_type	“string”	RAY_PARALLEL
r	real	–
r_eff	real	–

Generalized_Nat_Conv

Syntax

Bc Diffuse For Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Generalized_Nat_Conv ([Power_Output = *power_output*] | [Flux_Output = *flux_output*] | [Toggle = *toggle*] | [Multiplier = *multiplier*] | [H = *h*] | [T_Ref = *t_ref*])

Summary

Generalized natural convection energy flux

Description

Flux is given by:

$$F = Mh\left(T - T_{ref}\right)$$

M is given by MULTIPLIER, T_{ref} and h are taken from the BC REFERENCE TEMPERATURE and HEAT TRANSFER COEFFICIENT expressions.

The values for h and T_{ref} can also be supplied in-line as either constants or string functions.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output

If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Flux_Output

If specified, the average flux of the given BC/Src is calculated and stored into the global variable name specified as the *flux_output* argument.

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
power_output	“string”	–
flux_output	“string”	–
toggle	“string”	–
multiplier	real	1
h	“string”	–
t_ref	“string”	–

Generalized_Rad

Syntax

Bc Diffuse For Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Generalized_Rad ([Power_Output = *power_output*] | [Flux_Output = *flux_output*] | [Toggle = *toggle*] | [Multiplier = *multiplier*] | [T_Ref = *t_ref*])

Summary

Radiative energy flux, with values for emissivity, form factor and irradiation taken from expressions

Description

Flux is:

$$M \epsilon f \left(\sigma T^4 - I \right)$$

M is MULTIPLIER. σ is the Stefan-Boltzmann constant. ϵ and F are expressions for emissivity and form factor. T is surface temperature. I is the expression for irradiation. If I does not already exist, it will be constructed as σT_{ref}^4, where T_{ref} is either provided by the BC Rad Reference Temperature expression or the inline T_{ref} parameter (constant or string function).

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output

If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Flux_Output

If specified, the average flux of the given BC/Src is calculated and stored into the global variable name specified as the *flux_output* argument.

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
power_output	“string”	–
flux_output	“string”	–
toggle	“string”	–
multiplier	real	1
t_ref	“string”	–

Global

Syntax

Bc Diffuse For Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Power_Output = *power_output*] | [Flux_Output = *flux_output*] | [Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output

If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Flux_Output

If specified, the average flux of the given BC/Src is calculated and stored into the global variable name specified as the *flux_output* argument.

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
power_output	"string"	–
flux_output	"string"	–
toggle	"string"	–
global_name	"string"	–
component	integer	0

Kuntz

Syntax

Bc Diffuse For Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Kuntz ([Power_Output = *power_output*] | [Flux_Output = *flux_output*] | [Toggle = *toggle*] | [Multiplier = *multiplier*] | File = *file* | Units = *units*)

Summary

Interpolated value from tabulated data for the energy flux. FILE provides the filename for a table which specifies a flux value given X, Y, and Z coordinates and a temperature. UNITS specifies the units the table is in - ENGLISH ($\frac{BTU}{sft^2}$) or METRIC($\frac{W}{m^2}$)

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output

If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Flux_Output

If specified, the average flux of the given BC/Src is calculated and stored into the global variable name specified as the *flux_output* argument.

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
power_output	"string"	–
flux_output	"string"	–
toggle	"string"	–
multiplier	real	1
file	"string"	–
units	"string"	–

Lens

Syntax

Bc Diffuse For Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Lens ([Power_Output = *power_output*] | [Flux_Output = *flux_output*] | [Toggle = *toggle*] | Src_X = *src_x* | Dir_X = *dir_x* | Vel_X = *vel_x* | Src_Y = *src_y* | Dir_Y = *dir_y* | Vel_Y = *vel_y* | [Src_Z = *src_z*] | [Dir_Z = *dir_z*] | [Vel_Z = *vel_z*] | Massflowrate = *massFlowRate* | T_Liq = *T_liq* | T_Solid = *T_solid* | [T_On = *t_on*] | [T_Off = *t_off*] | Masssourceradius = *massSourceRadius* | Particletemperature = *particleTemperature* | Particledensity = *particleDensity*)

Summary

Energy flux for modeling the LENS process

Description

This boundary condition imposes an energy flux due to an incident mass source representing a powder particle stream due to the LENS process. The source is at the coordinates

$$\boldsymbol{s} = \boldsymbol{x}_s + \boldsymbol{v}_s(t - T_ON)$$

where $\boldsymbol{x}_s$ is provided by the SRC vector coordinates and $\boldsymbol{v}_s$ is provided by the VEL vector coordinates. The source is directed from by the DIR vector (). The

point-wise energy flux is computed using the equation below:

$$\frac{(-\boldsymbol{n})\cdot\boldsymbol{d}}{|\boldsymbol{d}|}\left(\rho c_p T\frac{\dot{m}}{\rho_p \pi R^2}-\frac{\dot{m}c_p T_p}{\pi R^2}\right)$$

ρ, c_p, and T are the density, specific heat, and temperature of the surface. R, $\dot{m}$, ρ_p, and T_p are given by the massSourceRadius, massFlowRate, particleDensity, and particleTemperature parameters. The term $-\boldsymbol{n}\cdot\boldsymbol{d}$ accounts for the fact that the surface may not be orthogonal to the source. For points where the minimum distance between the point and the line defined by and is greater than R, the flux will be set to zero. Additionally, if T is less than (T_liq+T_solid)/2, the flux will be set to zero. For times less than T_ON or greater than T_OFF, the flux will also be set to zero.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output
: If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Flux_Output
: If specified, the average flux of the given BC/Src is calculated and stored into the global variable name specified as the *flux_output* argument.

Toggle
: If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
power_output	“string”	–
flux_output	“string”	–
toggle	“string”	–
src_x	real	–
dir_x	real	–
vel_x	real	–
src_y	real	–
dir_y	real	–
vel_y	real	–
src_z	real	0
dir_z	real	0
vel_z	real	0
massFlowRate	real	–
T_liq	real	–
T_solid	real	–
t_on	real	0
t_off	real	1.798e+308
massSourceRadius	real	–
particleTemperature	real	–
particleDensity	real	–

Nat_Conv

Syntax

Bc Diffuse For Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nat_Conv ([Power_Output = *power_output*] | [Flux_Output = *flux_output*] | [Toggle = *toggle*] | H = *h* | T_Ref = *t_ref*)

Summary

Convective energy flux defined with constants specified for heat transfer coefficient and reference temperature.

Description

Flux is given by:

$$F = h\left(T - T_{ref}\right)$$

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output

If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Flux_Output

If specified, the average flux of the given BC/Src is calculated and stored into the global variable name specified as the *flux_output* argument.

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
power_output	“string”	–
flux_output	“string”	–
toggle	“string”	–
h	real	–
t_ref	real	–

Nd_Table

Syntax

Bc Diffuse For Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Power_Output = *power_output*] | [Flux_Output = *flux_output*] | [Toggle =

toggle] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table.

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output

If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Flux_Output

If specified, the average flux of the given BC/Src is calculated and stored into the global variable name specified as the *flux_output* argument.

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
power_output	“string”	–
flux_output	“string”	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Pid_Controlled

Syntax

Bc Diffuse For Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Pid_Controlled ([Power_Output = *power_output*] | [Flux_Output = *flux_output*] | [Toggle = *toggle*] | Max_Output = *Max_Output* | [P = *p*] | [I = *i*] | [D = *d*] | [T_I = *t_i*] | [T_D = *t_d*] | [Band = *Band*] | [Filter_Tau = *Filter_Tau*] | [Filter_Number = *Filter_Number*] | Setpoint_Function_Name = *Setpoint_Function_Name* | Control_Variable = *Control_Variable* | [Enable_Clamping = *enable_clamping*])

Summary

This flux term imposes an energy flux using a PID controller

Description

The controller setpoint vs. time is defined in a user function, and the controller feedback point uses a global variable (e.g. a data probe or reduction postprocessor).

The derivative term uses a filtered error value, using a low pass filter whose time constant is specified by the user. Like a heater, this is only capable of providing a positive influx. Even if the controller output is negative, the applied energy source will be truncated at 0. Likewise, the applied energy source cannot exceed the specified max source.

The nominal scaled PID controller output (between 0 and 1) is

$$u(t) = K_p e(t) + K_i \int_0^t e(s)ds + K_d \frac{\partial e_f}{\partial t}$$

where it is important to note that the K_p coefficient does not scale the others unless you have set it using the Band argument. In some PID implementations, the K_i and K_d coefficients are internally multiplied by K_p so care must be taken to provide the correct coefficients. This output $u(t)$ which is from 0 to 1 is scaled by the specified Max_Output value to produce the applied energy source. This means you **should not** include the output magnitude scaling in your P, I, or D parameters. If this controller is meant to recreate a controller that commands power directly, one must normalize by the max power.

When a derivative gain is provided, the derivative action is calculated using a filtered error to avoid derivative kick. The `Filter_Tau` argument can be used to specify the first-order derivative filter time constant τ, with a default value of 1 otherwise. Alternatively, one can provide the `Filter_Number` as argument $\tau = \frac{1}{N}$ where

$$K_d \frac{N}{1 + N\frac{1}{s}} = K_d \frac{1}{\tau + \frac{1}{s}} = \frac{K_d s}{\tau s + 1}$$

The P, I, and D coefficients can be specified in a number of ways:

- Just give a 'Band' argument. This calculates $P = 1/Band$ and uses the defaults of $I = 0.01P$ and $D = 0.0$
- Give a 'Band' argument and non-defaults for I and/or D. This still multiplies whatever you give for I and D by P.
- Give a 'P' argument instead of 'Band'. NOTE: in this mode the I and D values are not scaled by P.

If you provide both P and Band, P is used, the Band argument is ignored, and I and D are not scaled.

Alternatively, the integral and/or derivative gains can be provided using the integral time T_i (`T_i` argument) and derivative time T_d (`T_d` argument) so that $I = P/T_i$ and $D = P * T_d$ (regardless if P was set via band arg or not).

If `ENABLE_CLAMPING` is set to true, a clamping anti-windup algorithm is applied to the integral action of the controller.

The default values for I and D are 0.01 and 0 if not provided. The setpoint, controller output, error, filtered error, and filtered error time derivative are all output to automatically created global variables for diagnostics and postprocessing.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output
: If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Flux_Output
: If specified, the average flux of the given BC/Src is calculated and stored into the global variable name specified as the *flux_output* argument.

Toggle
: If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
power_output	“string”	–
flux_output	“string”	–
toggle	“string”	–
Max_Output	real	–
p	real	0
i	real	0
d	real	0
t_i	real	0
t_d	real	0
Band	real	0
Filter_Tau	real	1
Filter_Number	real	1
Setpoint_Function_Name	“string”	–
Control_Variable	“string”	–
enable_clamping	integer	–

Pid_Controlled_Bidirectional

Syntax

Bc Diffuse For Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Pid_Controlled_Bidirectional ([Power_Output = *power_output*] | [Flux_Output = *flux_output*] | [Toggle = *toggle*] | Max_Cooling_Power = *Max_Cooling_Power* | Max_Heating_Power = *Max_Heating_Power* | [P = *p*] | [I = *i*] | [D = *d*] | [T_I = *t_i*] | [T_D = *t_d*] | [Band = *Band*] | [Filter_Tau = *Filter_Tau*] | [Filter_Number = *Filter_Number*] | Setpoint_Function_Name = *Setpoint_Function_Name* | Control_Variable = *Control_Variable* | [Enable_Clamping = *enable_clamping*])

Summary

This is similar to PID_Controlled, however, the flux is applied to cool and heat the volume instead of just heating it. Max_Cooling_Power and Max_Heating_Power should be positive values. See description for PID_CONTROLLED for more details

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output

If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Flux_Output

If specified, the average flux of the given BC/Src is calculated and stored into the global variable name specified as the *flux_output* argument.

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
power_output	"string"	–
flux_output	"string"	–
toggle	"string"	–
Max_Cooling_Power	real	–
Max_Heating_Power	real	–
p	real	0
i	real	0
d	real	0
t_i	real	0
t_d	real	0
Band	real	0
Filter_Tau	real	1
Filter_Number	real	1
Setpoint_Function_Name	"string"	–
Control_Variable	"string"	–
enable_clamping	integer	–

Pid_Controlled_Cooler

Syntax

Bc Diffuse For Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Pid_Controlled_Cooler ([Power_Output = *power_output*] | [Flux_Output = *flux_output*] | [Toggle = *toggle*] | [P = *p*] | [I = *i*] | [D = *d*] | [T_I = *t_i*] | [T_D = *t_d*] | [Band = *Band*] | [Filter_Tau = *Filter_Tau*] | [Filter_Number = *Filter_Number*] | Setpoint_Function_Name = *Setpoint_Function_Name* | Control_Variable = *Control_Variable* | [Enable_Clamping = *enable_clamping*] | Max_Cooling_Power = *Max_Cooling_Power*)

Summary

This is similar to PID_Controlled, however, the flux is applied to cool the volume instead of heat it. Max_Cooling_Power should be a positive value. See description for PID_CONTROLLED for more details

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output

If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Flux_Output

If specified, the average flux of the given BC/Src is calculated and stored into the global variable name specified as the *flux_output* argument.

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
power_output	"string"	–
flux_output	"string"	–
toggle	"string"	–
p	real	0
i	real	0
d	real	0
t_i	real	0
t_d	real	0
Band	real	0
Filter_Tau	real	1
Filter_Number	real	1
Setpoint_Function_Name	"string"	–
Control_Variable	"string"	–
enable_clamping	integer	–
Max_Cooling_Power	real	–

Polynomial

Syntax

Bc Diffuse For Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Power_Output = *power_output*] | [Flux_Output = *flux_output*] | [Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The flux, F, is calculated as $F = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The

meaning of each option is as follows:

Power_Output
If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Flux_Output
If specified, the average flux of the given BC/Src is calculated and stored into the global variable name specified as the *flux_output* argument.

Toggle
If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
power_output	“string”	–
flux_output	“string”	–
toggle	“string”	–
variable	“string”	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Rad

Syntax

Bc Diffuse For Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Rad ([Power_Output = *power_output*] | [Flux_Output = *flux_output*] | [Toggle = *toggle*] | Crad = *crad* | T_Ref = *t_ref*)

Summary

Radiative energy flux defined with constant models for emissivity and reference temperature.

Description

The applied flux is given by:

$$c_{rad}\left(T^4 - T_{ref}^4\right)$$

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output

If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Flux_Output

If specified, the average flux of the given BC/Src is calculated and stored into the global variable name specified as the *flux_output* argument.

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
power_output	"string"	–
flux_output	"string"	–
toggle	"string"	–
crad	real	–
t_ref	real	–

Ray_Trace

Syntax

Bc Diffuse For Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Ray_Trace ([Power_Output = *power_output*] | [Flux_Output = *flux_output*] | [Toggle = *toggle*] | Ray_Source = *ray_source* | [Emission = *emission*])

Summary

Apply a ray tracing energy flux

Description

Adds an energy flux based on the intersection of rays defined in a ray tracing source block for the equation system with name given by the RAY_SOURCE parameter with the surface. The optional EMISSION parameter also imposes a σT^4 blackbody radiative emission term

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output

If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Flux_Output

If specified, the average flux of the given BC/Src is calculated and stored into the global variable name specified as the *flux_output* argument.

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
power_output	"string"	–
flux_output	"string"	–
toggle	"string"	–
ray_source	"string"	–
emission	"string"	FALSE

Rte_Field

Syntax

Bc Diffuse For Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Rte_Field ([Power_Output = *power_output*] | [Flux_Output = *flux_output*] | [Toggle = *toggle*])

Summary

RTE energy flux from a field

Description

Flux is given by -f, where f is the field named 'rte_radiative_flux'

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output

If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Flux_Output

If specified, the average flux of the given BC/Src is calculated and

stored into the global variable name specified as the *flux_output* argument.

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
power_output	“string”	–
flux_output	“string”	–
toggle	“string”	–

Rte_Sp

Syntax

Bc Diffuse For Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Rte_Sp ([Power_Output = *power_output*] | [Flux_Output = *flux_output*] | [Toggle = *toggle*] | Order = *order*)

Summary

RTE flux for energy equation with Simplified Spherical Harmonic RTE model with order ORDER

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output

If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Flux_Output

If specified, the average flux of the given BC/Src is calculated and stored into the global variable name specified as the *flux_output*

argument.

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
power_output	"string"	–
flux_output	"string"	–
toggle	"string"	–
order	integer	–

Scalar_String_Function

Syntax

Bc Diffuse For Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Power_Output = *power_output*] | [Flux_Output = *flux_output*] | [Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output

If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Flux_Output

If specified, the average flux of the given BC/Src is calculated and stored into the global variable name specified as the *flux_output* argument.

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
power_output	“string”	–
flux_output	“string”	–
toggle	“string”	–
f	“string”	–

Scaled_Nat_Conv

Syntax

Bc Diffuse For Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scaled_Nat_Conv ([Power_Output = *power_output*] | [Flux_Output = *flux_output*] | [Toggle = *toggle*] | [Multiplier = *multiplier*] | Scaling_Field = *scaling_field* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Scaled natural convection energy flux

Description

Flux is given by:

$$(M)(SC)(GV)(f)fcn(t)h\left(T-T_{ref}\right)$$

M is given by MULTIPLIER, SC by SCALING, GV by the value of the global variable with name GLOBAL_VAR, fcn is a function with name TIME_FUNCTION, and f is a field with name SCALING_FIELD. h and T_{ref} are taken from the HEAT TRANSFER COEFFICIENT and BC REFERENCE TEMPERATURE expressions.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output

If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Flux_Output

If specified, the average flux of the given BC/Src is calculated and stored into the global variable name specified as the *flux_output* argument.

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
power_output	“string”	–
flux_output	“string”	–
toggle	“string”	–
multiplier	real	1
scaling_field	“string”	–
scaling	real	1
global_var	“string”	–
time_function	“string”	–

Sharp_Power_Weld

Syntax

Bc Diffuse For Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Sharp_Power_Weld ([Power_Output = *power_output*] | [Flux_Output = *flux_output*] | [Toggle = *toggle*] | R = *r* | [Clip_Radius = *clip_radius*] | [Efficiency = *efficiency*] | Origin_And_Normal_Function = *origin_and_normal_function* | [Old_Temperature = *old_temperature*] | Path_Function = *path_function* | [Deposition = *deposition*] | [Integrated = *integrated*] | [Clip = *clip*])

Summary

Energy flux due to an incident 'top hat' beam

Description

The coordinates of the center of the beam are provided by the X, Y, and Z components of the provided origin_and_normal_function. The direction of the beam is provided by the NX, NY, and NZ components of the origin_and_normal_function. The applied flux is given by:

$$\frac{(-\boldsymbol{n}) \cdot \boldsymbol{d}}{|\boldsymbol{d}|} \frac{\alpha e P}{\pi R^2}$$

P is the laser power defined for the material, R is the beam radius, α is the laser absorptivity defined for the material, e is the EFFICIENCY, and the term $-\boldsymbol{n} \cdot \boldsymbol{d}$ accounts for the fact that the surface may not be orthogonal to the laser. Points with values of r_p > CLIP_RADIUS will receive zero flux. r_p is the shortest distance between a point on the surface and the line defined by the beam origin and normal. If ACTIVATION is set to true, points whose temperature exceed ACTIVATION_TEMPERATURE will be flagged as activated for use with the ACTIVATION USER FUNCTION thermal conductivity model. If INTEGRATED is true, this flux will be integrated between the previous time step and the current one and the average flux applied.

See the *Directed Energy* section for the appropriate syntax to use this flux in the directed energy block.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output

If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Flux_Output

If specified, the average flux of the given BC/Src is calculated and stored into the global variable name specified as the *flux_output* argument.

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
power_output	"string"	–
flux_output	"string"	–
toggle	"string"	–
r	real	–
clip_radius	real	0
efficiency	real	1
origin_and_normal_function	"string"	–
old_temperature	"string"	TRUE
path_function	"string"	–
deposition	"string"	FALSE
integrated	"string"	FALSE
clip	"string"	TRUE

Sharp_Ray_Trace

Syntax

Bc Diffuse For Energy ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Sharp_Ray_Trace ([Power_Output = *power_output*] | [Flux_Output = *flux_output*] | [Toggle = *toggle*] | Path_Function = *path_function* | Dir = *dir* | Power = *power* | Type = *type* | [N_Rays = *n_rays*] | [N_R = *n_r*] | [N_Theta_Initial = *n_theta_initial*] | [N_Intersections = *n_intersections*] | [Weight_Ratio = *weight_ratio*] | [Id = *id*] | [Absorption_Model = *absorption_model*] | [Parallel_Type = *parallel_type*] | R = *r*)

Summary

Ray tracing implementation of a sharp circular beam energy flux

Description

r - Laser spot radius.

type - Either radial discretization or Monte Carlo. Determines how beam will be discretized into rays. Radial discretization is deterministic and breaks beams into approximately equal area segments radially. Each segment generates one

ray. The power of each ray varies based on which segment it is generated from. Monte Carlo is stochastic and will generate a user-specified number of rays distributed spatially according to the laser's shape function, with each ray having equal power of power/n_rays.

n_r - Number of segments in radial direction for radial discretization.

n_theta_initial - Number of segments in theta direction in the smallest radial segment. The number of theta segments will grow as radial direction increases so segments have approximately the same area. Total number of rays will be n_r*n_r*n_theta_initial.

n_rays - Number of rays to generate for Monte Carlo discretization.

path_function - Function giving x, y, z coordinates of laser center as function of time.

dir - Function giving x,y,z components of direction that rays will travel initially from laser center defined by path_function.

power - Function giving total laser power.

n_intersections - Maximum number of times to intersect a ray with mesh elements due to reflections (default: 10).

weight_ratio - Fraction of initial ray power below which rays will be considered spent and no longer be traced (default: 0.001).

absorption_model - Either Specular, to use a surface emissivity value to determine fraction of energy absorbed and reflected with each ray intersection or Fresnel to use Fresnel model dependent on ray angle of incidence.

parallel_type - Either ray_parallel, to divide rays between processors with a globally shared mesh part or scene_parallel, to divide mesh part between processors with a globally shared set of rays.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output
: If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Flux_Output
: If specified, the average flux of the given BC/Src is calculated and stored into the global variable name specified as the *flux_output* argument.

Toggle
: If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
power_output	“string”	–
flux_output	“string”	–
toggle	“string”	–
path_function	“string”	–
dir	“string”	–
power	“string”	–
type	“string”	–
n_rays	integer	0
n_r	integer	0
n_theta_initial	integer	0
n_intersections	integer	10
weight_ratio	real	0.001
id	“string”	–
absorption_model	“string”	SPECULAR
parallel_type	“string”	RAY_PARALLEL
r	real	–

Thermal_Latent_Heat

Syntax

Bc Diffuse For Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Thermal_Latent_Heat ([Power_Output = *power_output*] | [Flux_Output = *flux_output*] | [Toggle = *toggle*])

Summary

Thermal latent heat energy flux

Description

Flux is given by:

$$H_v v_e$$

H_v is the expression for latent heat of vaporization and v_e is the local interface speed

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output
If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Flux_Output
If specified, the average flux of the given BC/Src is calculated and stored into the global variable name specified as the *flux_output* argument.

Toggle
If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
power_output	“string”	–
flux_output	“string”	–
toggle	“string”	–

User_Field

Syntax

Bc Diffuse For Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Power_Output = *power_output*] | [Flux_Output = *flux_output*] | [Toggle = *toggle*] | [Multiplier = *multiplier*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

User field energy flux

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output

If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Flux_Output

If specified, the average flux of the given BC/Src is calculated and stored into the global variable name specified as the *flux_output* argument.

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
power_output	“string”	–
flux_output	“string”	–
toggle	“string”	–
multiplier	real	1
name	“string”	–
scaling	real	1
global_var	“string”	–
time_function	“string”	–

User_Function

Syntax

Bc Diffuse For Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Power_Output = *power_output*] | [Flux_Output = *flux_output*] | [Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier =

multiplier])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output

If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Flux_Output

If specified, the average flux of the given BC/Src is calculated and stored into the global variable name specified as the *flux_output* argument.

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
power_output	“string”	–
flux_output	“string”	–
toggle	“string”	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc Diffuse For Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Power_Output = *power_output*] | [Flux_Output = *flux_output*] | [Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output

If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Flux_Output

If specified, the average flux of the given BC/Src is calculated and stored into the global variable name specified as the *flux_output* argument.

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
power_output	“string”	–
flux_output	“string”	–
toggle	“string”	–
Name	“string”	–
plugin_parameters	“string”	–

User_Vector_Field

Syntax

Bc Diffuse For Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Vector_Field ([Power_Output = *power_output*] | [Flux_Output = *flux_output*] | [Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*] | [Multiplier = *multiplier*])

Summary

User vector field energy flux

Description

Flux is given by:

$$(M)(SC)(GV)fcn(t)\boldsymbol{f}\cdot\boldsymbol{n}$$

M is given by MULTIPLIER, SC by SCALING, GV by the value of the global variable with name GLOBAL_VAR, fcn is a function with name TIME_FUNCTION, f is a field with name NAME, and n is the surface normal

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output

If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Flux_Output

If specified, the average flux of the given BC/Src is calculated and stored into the global variable name specified as the *flux_output* argument.

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
power_output	"string"	–
flux_output	"string"	–
toggle	"string"	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–
multiplier	real	1

User_Vector_Field_Influx

Syntax

Bc Diffuse For Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Vector_Field_Influx ([Power_Output = *power_output*] | [Flux_Output = *flux_output*] | [Toggle = *toggle*] | [Multiplier = *multiplier*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

User vector field influx energy flux

Description

Flux is given by:

$$(M)(SC)(GV)fcn(t)\boldsymbol{f}\cdot\boldsymbol{n}$$

M is given by MULTIPLIER, SC by SCALING, GV by the value of the global variable with name GLOBAL_VAR, fcn is a function with name TIME_FUNCTION, f is a field with name NAME, and n is the surface normal. Flux will be capped at zero (inflow only)

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output

If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Flux_Output

If specified, the average flux of the given BC/Src is calculated and stored into the global variable name specified as the *flux_output* argument.

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
power_output	“string”	–
flux_output	“string”	–
toggle	“string”	–
multiplier	real	1
name	“string”	–
scaling	real	1
global_var	“string”	–
time_function	“string”	–

Vapor_Cooling

Syntax

Bc Diffuse For Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Vapor_Cooling ([Power_Output = *power_output*] | [Flux_Output = *flux_output*] | [Toggle = *toggle*] | Tboil = *tboil*)

Summary

Empirical relation for cooling due to vaporization of stainless steel.

Description

Flux is given by:

$$814373(T - T_{boil}) - 2248.31(T - T_{boil})^2 + 27.1683(T - T_{boil})^3$$

For $T - T_{boil}$ less than 170 and:

$$-310360000 + 3.272400(T - T_{boil}) - 1808.4(T - T_{boil})^2 + 2.7284(T - T_{boil})^3$$

For $T - T_{boil}$ greater than 170. For $T - T_{boil}$ less than zero, no flux is applied

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output

If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Flux_Output

If specified, the average flux of the given BC/Src is calculated and stored into the global variable name specified as the *flux_output* argument.

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
power_output	“string”	–
flux_output	“string”	–
toggle	“string”	–
tboil	real	–

Enthalpy

Constant

Syntax

Bc Diffuse For Enthalpy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | Value = *value*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
value	real	–

Convective_Outflow

Syntax

Bc Diffuse For Enthalpy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Convective_Outflow [Toggle = *toggle*]

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–

Exponential

Syntax

Bc Diffuse For Enthalpy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Generalized_Nat_Conv

Syntax

Bc Diffuse For Enthalpy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Generalized_Nat_Conv ([Toggle = *toggle*] | [Multiplier = *multiplier*])

Summary

Generalized natural convection enthalpy flux

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
multiplier	real	1

Generalized_Rad

Syntax

Bc Diffuse For Enthalpy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Generalized_Rad ([Toggle = *toggle*] | [Multiplier = *multiplier*] | [T_Ref = *t_ref*])

Summary

Radiative energy flux, with values for emissivity, form factor and irradiation taken from expressions

Description

Flux is:

$$M\epsilon f\left(\sigma T^4 - I\right)$$

M is MULTIPLIER. σ is the Stefan-Boltzmann constant. ϵ and F are expressions for emissivity and form factor. T is surface temperature. I is the expression for irradiation. If I does not already exist, it will be constructed as σT_{ref}^4, where T_{ref} is either provided by the BC Rad Reference Temperature expression or the inline T_{ref} parameter (constant or string function).

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
multiplier	real	1
t_ref	“string”	–

Global

Syntax

Bc Diffuse For Enthalpy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
global_name	"string"	–
component	integer	0

Nat_Conv

Syntax

Bc Diffuse For Enthalpy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nat_Conv ([Toggle = *toggle*] | H = *h* | T_Ref = *t_ref*)

Summary

Natural convection BC for enthalpy flux

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
h	real	–
t_ref	real	–

Nd_Table

Syntax

Bc Diffuse For Enthalpy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc Diffuse For Enthalpy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
variable	“string”	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Rad

Syntax

Bc Diffuse For Enthalpy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Rad ([Toggle = *toggle*] | Crad = *crad* | T_Ref = *t_ref*)

Summary

Radiative energy flux defined with constant models for emissivity and reference temperature.

Description

The applied flux is given by:

$$c_{rad}\left(T^4 - T_{ref}^4\right)$$

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
crad	real	–
t_ref	real	–

Scalar_String_Function

Syntax

Bc Diffuse For Enthalpy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f	"string"	–

User_Field

Syntax

Bc Diffuse For Enthalpy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc Diffuse For Enthalpy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc Diffuse For Enthalpy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

Level_Set

Constant

Syntax

Bc Diffuse For Level_Set ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | *{flux | value}* = *flux*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
flux	real	–

Exponential

Syntax

Bc Diffuse For Level_Set ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Diffuse For Level_Set ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
global_name	"string"	–
component	integer	0

Inflow_Outflow

Syntax

Bc Diffuse For Level_Set ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Inflow_Outflow [Toggle = *toggle*]

Summary

This model has been deprecated and removed from the code.

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–

Nd_Table

Syntax

Bc Diffuse For Level_Set ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc Diffuse For Level_Set ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
variable	“string”	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Diffuse For Level_Set ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
f	“string”	–

User_Field

Syntax

Bc Diffuse For Level_Set ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc Diffuse For Level_Set ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc Diffuse For Level_Set ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

Lumped_Div_Projection

Constant

Syntax

Bc Diffuse For Lumped_Div_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | [Valuex = *valuex*] | [Valuey = *valuey*] | [Valuez = *valuez*])

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
valuex	real	0
valuey	real	0
valuez	real	0

Vector_String_Function

Syntax

Bc Diffuse For Lumped_Div_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Vector_String_Function ([Toggle = *toggle*] | F_X = *f_x* | F_Y = *f_y* | [F_Z = *f_z*])

Summary

Value from a string function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f_x	"string"	–
f_y	"string"	–
f_z	"string"	–

Lumped_Div_Projection_X

Constant

Syntax

Bc Diffuse For Lumped_Div_Projection_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | Value = *value*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
value	real	–

Exponential

Syntax

Bc Diffuse For Lumped_Div_Projection_X ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Diffuse For Lumped_Div_Projection_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Bc Diffuse For Lumped_Div_Projection_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc Diffuse For Lumped_Div_Projection_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Diffuse For Lumped_Div_Projection_X ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f	"string"	–

User_Field

Syntax

Bc Diffuse For Lumped_Div_Projection_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc Diffuse For Lumped_Div_Projection_X ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc Diffuse For Lumped_Div_Projection_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

Lumped_Div_Projection_Y

Constant

Syntax

Bc Diffuse For Lumped_Div_Projection_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | Value = *value*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
value	real	–

Exponential

Syntax

Bc Diffuse For Lumped_Div_Projection_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Diffuse For Lumped_Div_Projection_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Bc Diffuse For Lumped_Div_Projection_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc Diffuse For Lumped_Div_Projection_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Diffuse For Lumped_Div_Projection_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f	"string"	–

User_Field

Syntax

Bc Diffuse For Lumped_Div_Projection_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc Diffuse For Lumped_Div_Projection_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc Diffuse For Lumped_Div_Projection_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

Lumped_Div_Projection_Z

Constant

Syntax

Bc Diffuse For Lumped_Div_Projection_Z ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | Value = *value*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
value	real	–

Exponential

Syntax

Bc Diffuse For Lumped_Div_Projection_Z ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Diffuse For Lumped_Div_Projection_Z ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
global_name	“string”	–
component	integer	0

Nd_Table

Syntax

Bc Diffuse For Lumped_Div_Projection_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc Diffuse For Lumped_Div_Projection_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Diffuse For Lumped_Div_Projection_Z ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f	"string"	–

User_Field

Syntax

Bc Diffuse For Lumped_Div_Projection_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc Diffuse For Lumped_Div_Projection_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc Diffuse For Lumped_Div_Projection_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

Mass_Balance

Co2_Convective_Outflow

Syntax

Bc Diffuse For Mass_Balance ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Co2_Convective_Outflow ([Toggle = *toggle*] | [Ref_Frac = *ref_frac*])

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
ref_frac	real	0

Constant

Syntax

Bc Diffuse For Mass_Balance ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | Value = *value*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
value	real	–

Constant_Velocity

Syntax

Bc Diffuse For Mass_Balance ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant_Velocity ([Toggle = *toggle*] | Mult = *Mult*)

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is

active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Mult	real	–

Convective_Outflow

Syntax

Bc Diffuse For Mass_Balance ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Convective_Outflow ([Toggle = *toggle*] | [Ref_Frac = *ref_frac*])

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
ref_frac	real	0

Exponential

Syntax

Bc Diffuse For Mass_Balance ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
variable	“string”	–
constant	real	0
multiplier	real	1
exponent	real	–

Fluid_Robin_Coupled_One_Region

Syntax

Bc Diffuse For Mass_Balance ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Fluid_Robin_Coupled_One_Region ([Toggle = *toggle*] | [Coeff_Scaling = *coeff_scaling*])

Summary

Boundary condition for coupling of continuity to a free fluid in the same region.

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
coeff_scaling	real	0.001

Global

Syntax

Bc Diffuse For Mass_Balance ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The

meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
global_name	"string"	–
component	integer	0

Mass_Fluid_Robin_Coupled

Syntax

Bc Diffuse For Mass_Balance ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Mass_Fluid_Robin_Coupled ([Toggle = *toggle*] | [Coeff_Scaling = *coeff_scaling*] | P_Field = *P_Field* | Massflux_Field = *MassFlux_Field*)

Summary

Boundary condition for loose coupling of continuity to a free fluid region

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
coeff_scaling	real	0.001
P_Field	"string"	–
MassFlux_Field	"string"	–

Mass_Fraction_Fluid_Robin_Coupled

Syntax

Bc Diffuse For Mass_Balance ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Mass_Fraction_Fluid_Robin_Coupled ([Toggle = *toggle*] | Mass_Fraction_Field = *mass_fraction_field* | Mass_Fraction_Diffusive_Flux_Field = *mass_fraction_diffusive_flux_field* | Coeff_Field = *coeff_field*)

Summary

Boundary condition for loose coupling of scalars to a free fluid region

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
mass_fraction_field	"string"	–
mass_fraction_diffusive_flux_field	"string"	–
coeff_field	"string"	–

Mass_Fraction_Fluid_Robin_Coupled_One_Region

Syntax

Bc Diffuse For Mass_Balance ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Mass_Fraction_Fluid_Robin_Coupled_One_Region [Toggle = *toggle*]

Summary

Boundary condition for coupling of scalars to a free fluid in the same region.

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–

Mass_Open

Syntax

Bc Diffuse For Mass_Balance ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Mass_Open ([Toggle = *toggle*] | [Pressure = *pressure*])

Summary

A weakly enforced pressure condition for open flow from a porous flow domain

Description

This model weakly enforces a pressure condition on the boundary of a porous domain, either equal to the specified constant `PRESSURE`, or set using the `BOUNDARY_PRESSURE` model from the surface material.

Note: This should only be used with porous flow equations.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
pressure	real	0

Mesh_Motion

Syntax

Bc Diffuse For Mass_Balance ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Mesh_Motion [Toggle = *toggle*]

Summary

Moving boundary condition for Arbitrary Lagrangian-Eulerian (ALE) mesh motion in relative flux form. Accounts for the boundary advective flux due to general mesh motion.

Description

Applies the following boundary flux

$$-\int q\hat{\mathbf{u}}\cdot\mathbf{n}dS,$$

where $\hat{u}$ is the mesh velocity and where the integrated scalar q depends on the equation. For the gas phase `porous_enthalpy` equation with a `transported enthalpy = porous`, $q = \phi\rho_g h_g$. Similarly, for the gas phase `mass_balance` equation, $q = \phi\rho_g$ is the bulk mass density.

This flux boundary condition is automatically applied to the mass balance and porous enthalpy equations when the `use relative mass flux` porous flow option is turned on.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–

Moving_Boundary

Syntax

Bc Diffuse For Mass_Balance ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Moving_Boundary [Toggle = *toggle*]

Summary

No penetration moving boundary condition, valid for Lagrangian mesh motion in non-relative flux form or relative flux form.

Description

Applies the following boundary flux

$$\int q\hat{\mathbf{u}} \cdot \mathbf{n} dS,$$

where $\hat{u}$ is the mesh velocity and where the integrated scalar q depends on the equation. If we consider the gas phase `porous_enthalpy` equation with a `transported enthalpy = porous`, then $q = \phi\rho_g h_g$. Similarly, for the gas phase `mass_balance` equation, $q = \phi\rho_g$ is the bulk mass density.

If we consider the gas phase `mass_balance` equation, with a gas phase porous velocity relative to the moving solid phase, then physically this boundary condition is derived from applying a no flux condition at moving boundary i.e.,

$$\mathbf{u}_g \cdot \mathbf{n} = \phi\hat{\mathbf{u}} \cdot \mathbf{n}.$$

In the Lagrangian setting, this flux is required to impose a no flux condition at the moving boundary with both the non-relative form (default) and relative form of the advection term. When using the relative form, typically a do-nothing condition would be sufficient to impose no flux; however, to simplify general ALE mesh motion for porous flow equations (i.e., mass balance and porous enthalpy) we automatically provide the mesh motion flux on boundaries for blocks that have mesh motion. Consequently, to impose a zero flux, this term is needed for cancellation.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–

Nd_Table

Syntax

Bc Diffuse For Mass_Balance ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Open

Syntax

Bc Diffuse For Mass_Balance ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Open ([Toggle = *toggle*] | [Pressure = *pressure*] | [Entrained_Value = *entrained_value*])

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
pressure	real	0
entrained_value	real	0

Open_Flow

Syntax

Bc Diffuse For Mass_Balance ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Open_Flow [Toggle = *toggle*]

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–

Polynomial

Syntax

Bc Diffuse For Mass_Balance ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
variable	“string”	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Diffuse For Mass_Balance ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
f	“string”	–

User_Field

Syntax

Bc Diffuse For Mass_Balance ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
scaling	real	1
global_var	“string”	–
time_function	“string”	–

User_Function

Syntax

Bc Diffuse For Mass_Balance ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc Diffuse For Mass_Balance ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

Well_Outflow

Syntax

Bc Diffuse For Mass_Balance ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Well_Outflow ([Toggle = *toggle*] | Pi = *pi* | Pref = *pref*)

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
pi	"string"	–
pref	"string"	–

Mesh

Constant

Syntax

Bc Diffuse For Mesh ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | [Valuex = *valuex*] | [Valuey = *valuey*] | [Valuez = *valuez*])

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
valuex	real	0
valuey	real	0
valuez	real	0

Constant_Traction

Syntax

Bc Diffuse For Mesh ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant_Traction ([Toggle = *toggle*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Constant traction mesh flux

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
x	real	0
y	real	0
z	real	0

Damping

Syntax

Bc Diffuse For Mesh ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Damping ([Toggle = *toggle*] | [Scale_Factor = *scale_factor*])

Summary

Dampen surface displacements.

Description

$$F = \frac{-2\mu s}{h}\frac{\partial u}{\partial t}$$

where 2μ is the Lame constant, s the optional "scale_factor" parameter, h the mesh size, and u the mesh displacements. This BC is primarily intended for use in conjunction with a rotated BC that enforces the normal mesh displacement, and serves to dampen tangential mesh displacements on the surface.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
scale_factor	real	1

Electric_Traction

Syntax

Bc Diffuse For Mesh ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Electric_Traction ([Toggle = *toggle*] | [Sign = *sign*])

Summary

Electric traction mesh flux

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
sign	real	1

Penalized_Normal_Displacement

Syntax

Bc Diffuse For Mesh ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Penalized_Normal_Displacement ([Toggle = *toggle*] | [Stiffness_Factor = *stiffness_factor*])

Summary

Penalized normal flux

Description

Enforces a weak zero-normal-displacement condition by applying a mesh flux of

$$\vec{F} = -2\mu S_f \left(\vec{D} \cdot \vec{n}\right) \vec{n}$$

Where 2μ is the mesh `TWO_MU` value, and S_f is the stiffness factor.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
stiffness_factor	real	1e+06

Pressure

Syntax

Bc Diffuse For Mesh ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Pressure ([Toggle = *toggle*] | [User_Field_Name = *user_field_name*] | [Use_Bulk_Node = *use_bulk_node*] | [P = *p*] | [C_T = *c_t*] | [C_X = *c_x*] | [C_Y = *c_y*] | [C_Z = *c_z*] | [C_Cos_W = *c_cos_w*] | [C_Sin_W = *c_sin_w*] | [C1_Cos = *c1_cos*] | [C1_Sin = *c1_sin*])

Summary

Pressure mesh flux

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
user_field_name	"string"	–
use_bulk_node	"string"	–
p	real	0
c_t	real	0
c_x	real	0
c_y	real	0
c_z	real	0
c_cos_w	real	0
c_sin_w	real	0
c1_cos	real	0
c1_sin	real	0

Pressure_User_Function

Syntax

Bc Diffuse For Mesh ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Pressure_User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Pressure user function mesh flux

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1

Shear_Free

Syntax

Bc Diffuse For Mesh ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Shear_Free [Toggle = *toggle*]

Summary

Shear-free traction mesh flux

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–

Specified_Normal_Velocity

Syntax

Bc Diffuse For Mesh ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Specified_Normal_Velocity ([Toggle = *toggle*] | [Stiffness_Factor = *stiffness_factor*])

Summary

Penalized normal mesh velocity

Description

Enforces a weak specified normal velocity condition by applying a mesh flux of

$$\vec{F} = 2\mu S_f \left(V_m - \dot{\vec{D}} \cdot \vec{n} \right) \vec{n}$$

Where 2μ is the mesh `TWO_MU` value, and S_f is the stiffness factor. The normal mesh velocity, V_m is provided by the `NORMAL_MESH_VELOCITY` expression on the surface (scalar, positive for outward mesh velocity).

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
stiffness_factor	real	1e+06

Transient_Traction

Syntax

Bc Diffuse For Mesh ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Transient_Traction ([Toggle = *toggle*] | [A_X = *a_x*] | [A_Y = *a_y*] | [A_Z = *a_z*] | [B_X = *b_x*] | [B_Y = *b_y*] | [B_Z = *b_z*])

Summary

Transient traction mesh flux

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
a_x	real	0
a_y	real	0
a_z	real	0
b_x	real	0
b_y	real	0
b_z	real	0

Vector_String_Function

Syntax

Bc Diffuse For Mesh ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Vector_String_Function ([Toggle = *toggle*] | F_X = *f_x* | F_Y = *f_y* | [F_Z = *f_z*])

Summary

Value from a string function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f_x	"string"	–
f_y	"string"	–
f_z	"string"	–

Mesh_X

Constant

Syntax

Bc Diffuse For Mesh_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | Value = *value*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
value	real	–

Exponential

Syntax

Bc Diffuse For Mesh_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Diffuse For Mesh_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
global_name	“string”	–
component	integer	0

Nd_Table

Syntax

Bc Diffuse For Mesh_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc Diffuse For Mesh_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Diffuse For Mesh_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f	"string"	–

User_Field

Syntax

Bc Diffuse For Mesh_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc Diffuse For Mesh_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc Diffuse For Mesh_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

Mesh_Y

Constant

Syntax

Bc Diffuse For Mesh_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | Value = *value*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
value	real	–

Exponential

Syntax

Bc Diffuse For Mesh_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Diffuse For Mesh_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Bc Diffuse For Mesh_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc Diffuse For Mesh_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Diffuse For Mesh_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f	"string"	–

User_Field

Syntax

Bc Diffuse For Mesh_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc Diffuse For Mesh_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc Diffuse For Mesh_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
Name	“string”	–
plugin_parameters	“string”	–

Mesh_Z

Constant

Syntax

Bc Diffuse For Mesh_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | Value = *value*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
value	real	–

Exponential

Syntax

Bc Diffuse For Mesh_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Diffuse For Mesh_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
global_name	“string”	–
component	integer	0

Nd_Table

Syntax

Bc Diffuse For Mesh_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc Diffuse For Mesh_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Diffuse For Mesh_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f	"string"	–

User_Field

Syntax

Bc Diffuse For Mesh_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc Diffuse For Mesh_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc Diffuse For Mesh_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

Mixture_Fraction

Constant

Syntax

Bc Diffuse For Mixture_Fraction ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | *{flux | value}* = *flux*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
flux	real	–

Exponential

Syntax

Bc Diffuse For Mixture_Fraction ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Diffuse For Mixture_Fraction ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Bc Diffuse For Mixture_Fraction ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Open_Flow_Convection

Syntax

Bc Diffuse For Mixture_Fraction ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Open_Flow_Convection ([Toggle = *toggle*] | Reference_Value = *Reference_Value*)

Summary

Open flow convection BC for the mixture fraction equation

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Reference_Value	real	–

Polynomial

Syntax

Bc Diffuse For Mixture_Fraction ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Diffuse For Mixture_Fraction ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f	"string"	–

User_Field

Syntax

Bc Diffuse For Mixture_Fraction ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
scaling	real	1
global_var	“string”	–
time_function	“string”	–

User_Function

Syntax

Bc Diffuse For Mixture_Fraction ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc Diffuse For Mixture_Fraction ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

Momentum

Capillary

Syntax

Bc Diffuse For Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Capillary [Toggle = *toggle*]

Summary

LS capillary momentum flux

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–

Capillary_Gap

Syntax

Bc Diffuse For Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Capillary_Gap ([Toggle = *toggle*] | V_W = *v_w* | G = *g* | Theta = *theta*)

Summary

This model has been deprecated

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
v_w	real	–
g	real	–
theta	real	–

Constant

Syntax

Bc Diffuse For Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
x	real	0
y	real	0
z	real	0

Constant_Traction

Syntax

Bc Diffuse For Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant_Traction ([Toggle = *toggle*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
x	real	0
y	real	0
z	real	0

Darcy_Slip

Syntax

Bc Diffuse For Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Darcy_Slip ([Toggle = *toggle*] | [Vs_X = *vs_x*] | [Vs_Y = *vs_y*] | [Vs_Z = *vs_z*] | K = *k*)

Summary

Darcy slip momentum flux

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
vs_x	real	0
vs_y	real	0
vs_z	real	0
k	real	–

Electric_Traction

Syntax

Bc Diffuse For Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Electric_Traction ([Toggle = *toggle*] | [Sign = *sign*])

Summary

Electric traction momentum flux

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
sign	real	1

Equilibrium_Capillary_Gap

Syntax

Bc Diffuse For Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Equilibrium_Capillary_Gap ([Toggle = *toggle*] | Theta = *theta*)

Summary

This model has been deprecated

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
theta	real	–

Exponential_Vapor_Recoil_Pressure

Syntax

Bc Diffuse For Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential_Vapor_Recoil_Pressure ([Toggle = *toggle*] | Tboil = *tboil* | L = *l* | Pa = *pa* | [Beta = *beta*] | Mw = *mw* | [Tl = *tl*] | [Th = *th*])

Summary

Exponential vapor recoil pressure momentum flux

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
tboil	real	–
l	real	–
pa	real	–
beta	real	1
mw	real	–
tl	real	-1
th	real	-1

Fixed_Curvature

Syntax

Bc Diffuse For Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Fixed_Curvature ([Toggle = *toggle*] | Curvature = *curvature*)

Summary

This model has been deprecated and removed from the code.

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
curvature	real	–

Flow_Hydrostatic

Syntax

Bc Diffuse For Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Flow_Hydrostatic ([Toggle = *toggle*] | [P_Ref = *p_ref*] | [Gx = *gx*] | [Gy = *gy*] | [Gz = *gz*])

Summary

Flow hydrostatic momentum flux

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
p_ref	real	0
gx	real	0
gy	real	0
gz	real	0

Free_Open_Flow

Syntax

Bc Diffuse For Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Free_Open_Flow ([Toggle = *toggle*] | Pressure = *pressure*)

Summary

Open flow for momentum; apply the full stress

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
pressure	real	–

Lens_Particles_Pressure

Syntax

Bc Diffuse For Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Lens_Particles_Pressure ([Toggle = *toggle*] | Src_X = *src_x* | Dir_X = *dir_x* | Vel_X = *vel_x* | Src_Y = *src_y* | Dir_Y = *dir_y* | Vel_Y = *vel_y* | [Src_Z = *src_z*] | [Dir_Z = *dir_z*] | [Vel_Z = *vel_z*] | Massflowrate = *massFlowRate* | T_Liq = *T_liq* | T_Solid = *T_solid* | [T_On = *t_on*] | [T_Off = *t_off*] | Particlespeed = *particleSpeed* | Radius = *radius*)

Summary

LENS particles pressure momentum flux

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
src_x	real	–
dir_x	real	–
vel_x	real	–
src_y	real	–
dir_y	real	–
vel_y	real	–
src_z	real	0
dir_z	real	0
vel_z	real	0
massFlowRate	real	–
T_liq	real	–
T_solid	real	–
t_on	real	0
t_off	real	1.798e+308
particleSpeed	real	–
radius	real	–

Ls_Capillary

Syntax

Bc Diffuse For Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Ls_Capillary [Toggle = *toggle*]

Summary

LS capillary momentum flux

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–

Ls_Oriented_Slip

Syntax

Bc Diffuse For Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Ls_Oriented_Slip ([Toggle = *toggle*] | [Vs_X = *vs_x*] | [Vs_Y = *vs_y*] | [Vs_Z = *vs_z*] | [Beta_Normal_A = *beta_normal_a*] | [Beta_Tangent_A = *beta_tangent_a*] | [Beta_Normal_B = *beta_normal_b*] | [Beta_Tangent_B = *beta_tangent_b*])

Summary

Oriented slip momentum flux

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
vs_x	real	0
vs_y	real	0
vs_z	real	0
beta_normal_a	real	0
beta_tangent_a	real	0
beta_normal_b	real	0
beta_tangent_b	real	0

Node_Normal_Capillary

Syntax

Bc Diffuse For Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Node_Normal_Capillary [Toggle = *toggle*]

Summary

This model has been deprecated and removed from the code.

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–

Non_Ibp_Pressure

Syntax

Bc Diffuse For Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Non_Ibp_Pressure ([Toggle = *toggle*] | [Pressure = *pressure*])

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
pressure	"string"	–

Open_Flow

Syntax

Bc Diffuse For Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Open_Flow ([Toggle = *toggle*] | [Pressure = *pressure*] | [Total_Pressure = *total_pressure*])

Summary

Open flow for momentum; enforce normal gradient to be zero

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
pressure	real	0
total_pressure	real	0

Oriented_Slip

Syntax

Bc Diffuse For Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Oriented_Slip ([Toggle = *toggle*] | [Vs_X = *vs_x*] | [Vs_Y = *vs_y*] | [Vs_Z = *vs_z*] | [Vs_Expr = *vs_expr*] | Beta_Normal = *beta_normal* | Beta_Tangent = *beta_tangent*)

Summary

Oriented slip momentum flux. Essentially a sum of the Navier-Slip condition and a penalty condition for enforcing velocity in the normal direction.

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
vs_x	real	0
vs_y	real	0
vs_z	real	0
vs_expr	"string"	–
beta_normal	real	–
beta_tangent	real	–

Pressure

Syntax

Bc Diffuse For Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Pressure ([Toggle = *toggle*] | [User_Field_Name = *user_field_name*] | [Use_Bulk_Node = *use_bulk_node*] | [P = *p*] | [C_T = *c_t*] | [C_X = *c_x*] | [C_Y = *c_y*] | [C_Z = *c_z*] | [C_Cos_W = *c_cos_w*] | [C_Sin_W = *c_sin_w*] | [C1_Cos = *c1_cos*] | [C1_Sin = *c1_sin*])

Summary

Pressure momentum flux

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is

active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
user_field_name	"string"	–
use_bulk_node	"string"	–
p	real	0
c_t	real	0
c_x	real	0
c_y	real	0
c_z	real	0
c_cos_w	real	0
c_sin_w	real	0
c1_cos	real	0
c1_sin	real	0

Pressure_User_Function

Syntax

Bc Diffuse For Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Pressure_User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Pressure user function momentum flux

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1

Projected_Capillary

Syntax

Bc Diffuse For Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Projected_Capillary [Toggle = *toggle*]

Summary

Projected capillary momentum flux

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–

Reconstructed_Curvature

Syntax

Bc Diffuse For Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Reconstructed_Curvature ([Toggle = *toggle*] | [Ls_Size = *ls_size*] | [Fixed_Curvature = *fixed_curvature*])

Summary

This model has been deprecated and removed from the code.

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
ls_size	integer	0
fixed_curvature	real	-1

Simple_Spring_Force

Syntax

Bc Diffuse For Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Simple_Spring_Force ([Toggle = *toggle*] | [Kx = *kx*] | [Ky = *ky*] | [Kz = *kz*] | [X0 = *x0*] | [Y0 = *y0*] | [Z0 = *z0*])

Summary

This model has been deprecated and removed from the code.

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
kx	real	0
ky	real	0
kz	real	0
x0	real	0
y0	real	0
z0	real	0

Slip

Syntax

Bc Diffuse For Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Slip ([Toggle = *toggle*] | [Vs_X = *vs_x*] | [Vs_Y = *vs_y*] | [Vs_Z = *vs_z*] | Beta = *beta*)

Summary

Slip momentum flux

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
vs_x	real	0
vs_y	real	0
vs_z	real	0
beta	real	–

Slip_Length

Syntax

Bc Diffuse For Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Slip_Length ([Toggle = *toggle*] | [Vs_X = *vs_x*] | [Vs_Y = *vs_y*] | [Vs_Z = *vs_z*] | Beta = *beta*)

Summary

Slip length momentum flux

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
vs_x	real	0
vs_y	real	0
vs_z	real	0
beta	real	–

Spring_Force

Syntax

Bc Diffuse For Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Spring_Force ([Toggle = *toggle*] | Mult = *Mult*)

Summary

Spring force for momentum boundary condition

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Mult	real	–

Symmetry_Flow

Syntax

Bc Diffuse For Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Symmetry_Flow [Toggle = *toggle*]

Summary

Symmetry flow momentum flux

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–

Transient_Traction

Syntax

Bc Diffuse For Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Transient_Traction ([Toggle = *toggle*] | [A_X = *a_x*] | [A_Y = *a_y*] | [A_Z = *a_z*] | [B_X = *b_x*] | [B_Y = *b_y*] | [B_Z = *b_z*])

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
a_x	real	0
a_y	real	0
a_z	real	0
b_x	real	0
b_y	real	0
b_z	real	0

Vapor_Recoil_Pressure

Syntax

Bc Diffuse For Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Vapor_Recoil_Pressure ([Toggle = *toggle*] | Tboil = *tboil*)

Summary

Vapor recoil pressure momentum flux

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
tboil	real	–

Vector_String_Function

Syntax

Bc Diffuse For Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Vector_String_Function ([Toggle = *toggle*] | F_X = *f_x* | F_Y = *f_y* | [F_Z = *f_z*])

Summary

Value from a string function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f_x	"string"	–
f_y	"string"	–
f_z	"string"	–

Momentum_X

Constant

Syntax

Bc Diffuse For Momentum_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | Value = *value*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
value	real	–

Exponential

Syntax

Bc Diffuse For Momentum_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Diffuse For Momentum_X ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
global_name	“string”	–
component	integer	0

Nd_Table

Syntax

Bc Diffuse For Momentum_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc Diffuse For Momentum_X ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Diffuse For Momentum_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f	"string"	–

User_Field

Syntax

Bc Diffuse For Momentum_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc Diffuse For Momentum_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc Diffuse For Momentum_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

Momentum_Y

Constant

Syntax

Bc Diffuse For Momentum_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | Value = *value*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
value	real	–

Exponential

Syntax

Bc Diffuse For Momentum_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Diffuse For Momentum_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
global_name	“string”	–
component	integer	0

Nd_Table

Syntax

Bc Diffuse For Momentum_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc Diffuse For Momentum_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Diffuse For Momentum_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f	"string"	–

User_Field

Syntax

Bc Diffuse For Momentum_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc Diffuse For Momentum_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc Diffuse For Momentum_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

Momentum_Z

Constant

Syntax

Bc Diffuse For Momentum_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | Value = *value*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
value	real	–

Exponential

Syntax

Bc Diffuse For Momentum_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Diffuse For Momentum_Z ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
global_name	“string”	–
component	integer	0

Nd_Table

Syntax

Bc Diffuse For Momentum_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc Diffuse For Momentum_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Diffuse For Momentum_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f	"string"	–

User_Field

Syntax

Bc Diffuse For Momentum_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc Diffuse For Momentum_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc Diffuse For Momentum_Z ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

Porous_Enthalpy

Constant

Syntax

Bc Diffuse For Porous_Enthalpy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | Value = *value*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
value	real	–

Convective_Outflow

Syntax

Bc Diffuse For Porous_Enthalpy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Convective_Outflow ([Toggle = *toggle*] | [Ref_H = *ref_h*])

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
ref_h	real	0

Enclosure_Radiation

Syntax

Bc Diffuse For Porous_Enthalpy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Enclosure_Radiation ([Toggle = *toggle*] | [Multiplier = *multiplier*] | Enclosure = *enclosure*)

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
multiplier	real	1
enclosure	“string”	–

Exponential

Syntax

Bc Diffuse For Porous_Enthalpy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the

constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Fluid_Robin_Coupled

Syntax

Bc Diffuse For Porous_Enthalpy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Fluid_Robin_Coupled ([Toggle = *toggle*] | Enthalpy_Field = *enthalpy_field* | Enthalpy_Diffusive_Flux_Field = *enthalpy_diffusive_flux_field* | Coeff_Field = *coeff_field*)

Summary

Boundary condition for loose coupling of enthalpy to a free fluid region

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
enthalpy_field	"string"	–
enthalpy_diffusive_flux_field	"string"	–
coeff_field	"string"	–

Fluid_Robin_Coupled_One_Region

Syntax

Bc Diffuse For Porous_Enthalpy ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Fluid_Robin_Coupled_One_Region [Toggle = *toggle*]

Summary

Boundary condition for loose coupling of enthalpy to a free fluid region

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–

Fluid_Robin_Coupled_With_Solid_Convection

Syntax

Bc Diffuse For Porous_Enthalpy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Fluid_Robin_Coupled_With_Solid_Convection ([Toggle = *toggle*] | Enthalpy_Field = *enthalpy_field* | Enthalpy_Diffusive_Flux_Field = *enthalpy_diffusive_flux_field* | Coeff_Field = *coeff_field*)

Summary

Boundary condition for loose coupling of enthalpy to a free fluid region

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
enthalpy_field	“string”	–
enthalpy_diffusive_flux_field	“string”	–
coeff_field	“string”	–

Fluid_Robin_Coupled_With_Solid_Convection_One_Region

Syntax

Bc Diffuse For Porous_Enthalpy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Fluid_Robin_Coupled_With_Solid_Convection_One_Region [Toggle = *toggle*]

Summary

Boundary condition for loose coupling of enthalpy to a free fluid region

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–

Fluid_Solid_Convection_Coupled

Syntax

Bc Diffuse For Porous_Enthalpy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Fluid_Solid_Convection_Coupled ([Toggle = *toggle*] | [Multiplier = *multiplier*] | Fluid_Temperature_Field = *fluid_temperature_field*)

Summary

Boundary condition for loose coupling of enthalpy to a free fluid region

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
multiplier	real	1
fluid_temperature_field	"string"	–

Fluid_Solid_Convection_Coupled_One_Region

Syntax

Bc Diffuse For Porous_Enthalpy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Fluid_Solid_Convection_Coupled_One_Region ([Toggle = *toggle*] | [Multiplier = *multiplier*])

Summary

Boundary condition for loose coupling of enthalpy to a free fluid region

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
multiplier	real	1

Generalized_Nat_Conv

Syntax

Bc Diffuse For Porous_Enthalpy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Generalized_Nat_Conv ([Toggle = *toggle*] | [Multiplier = *multiplier*])

Summary

Generalized natural convection flux BC

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
multiplier	real	1

Generalized_Rad

Syntax

Bc Diffuse For Porous_Enthalpy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Generalized_Rad ([Toggle = *toggle*] | [Multiplier = *multiplier*] | [T_Ref = *t_ref*])

Summary

Radiative energy flux, with values for emissivity, form factor and irradiation taken from expressions

Description

Flux is:

$$M\epsilon f\left(\sigma T^4 - I\right)$$

M is MULTIPLIER. σ is the Stefan-Boltzmann constant. ϵ and F are expressions for emissivity and form factor. T is surface temperature. I is the expression for irradiation. If I does not already exist, it will be constructed as σT_{ref}^4, where T_{ref} is either provided by the BC Rad Reference Temperature expression or the inline T_{ref} parameter (constant or string function).

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
multiplier	real	1
t_ref	“string”	–

Global

Syntax

Bc Diffuse For Porous_Enthalpy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
global_name	“string”	–
component	integer	0

Material_Phase_Sum_Open

Syntax

Bc Diffuse For Porous_Enthalpy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Material_Phase_Sum_Open ([Toggle = *toggle*] | [Pressure = *pressure*] | [Entrained_Value = *entrained_value*])

Summary

Open boundary condition for a single temperature model, wherein the phases are assumed to be in thermal equilibrium. The advective flux of enthalpy for

each phase is summed at the open.Must provide the BOUNDARY_ENTRAINED_ENTHALPY expression for entrainment.

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
pressure	real	0
entrained_value	real	0

Mesh_Motion

Syntax

Bc Diffuse For Porous_Enthalpy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Mesh_Motion [Toggle = *toggle*]

Summary

Moving boundary condition for Arbitrary Lagrangian-Eulerian (ALE) mesh motion in relative flux form. Accounts for the boundary advective flux due to general mesh motion.

Description

Applies the following boundary flux

$$-\int q\hat{\mathbf{u}}\cdot\mathbf{n}dS,$$

where $\hat{u}$ is the mesh velocity and where the integrated scalar q depends on the equation. For the gas phase `porous_enthalpy` equation with a `transported`

`enthalpy = porous`, $q = \phi\rho_g h_g$. Similarly, for the gas phase `mass_balance` equation, $q = \phi\rho_g$ is the bulk mass density.

This flux boundary condition is automatically applied to the mass balance and porous enthalpy equations when the `use relative mass flux` porous flow option is turned on.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle
If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–

Moving_Boundary

Syntax

Bc Diffuse For Porous_Enthalpy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Moving_Boundary [Toggle = *toggle*]

Summary

No penetration moving boundary condition, valid for Lagrangian mesh motion in non-relative flux form or relative flux form.

Description

Applies the following boundary flux

$$\int q\hat{\mathbf{u}} \cdot \mathbf{n} dS,$$

where $\hat{u}$ is the mesh velocity and where the integrated scalar q depends on the equation. If we consider the gas phase `porous_enthalpy` equation with a `transported enthalpy = porous`, then $q = \phi\rho_g h_g$. Similarly, for the gas phase `mass_balance` equation, $q = \phi\rho_g$ is the bulk mass density.

If we consider the gas phase `mass_balance` equation, with a gas phase porous velocity relative to the moving solid phase, then physically this boundary condition is derived from applying a no flux condition at moving boundary i.e.,

$$\mathbf{u}_g \cdot \mathbf{n} = \phi \hat{\mathbf{u}} \cdot \mathbf{n}.$$

In the Lagrangian setting, this flux is required to impose a no flux condition at the moving boundary with both the non-relative form (default) and relative form of the advection term. When using the relative form, typically a do-nothing condition would be sufficient to impose no flux; however, to simplify general ALE mesh motion for porous flow equations (i.e., mass balance and porous enthalpy) we automatically provide the mesh motion flux on boundaries for blocks that have mesh motion. Consequently, to impose a zero flux, this term is needed for cancellation.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–

Nd_Table

Syntax

Bc Diffuse For Porous_Enthalpy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

One_Temp_Multiphase_Open

Syntax

Bc Diffuse For Porous_Enthalpy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = One_Temp_Multiphase_Open ([Toggle = *toggle*] | [Pressure = *pressure*] | [Entrained_Value = *entrained_value*])

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
pressure	real	0
entrained_value	real	0

Open

Syntax

Bc Diffuse For Porous_Enthalpy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Open ([Toggle = *toggle*] | [Pressure = *pressure*] | [Entrained_Value = *entrained_value*])

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
pressure	real	0
entrained_value	real	0

Polynomial

Syntax

Bc Diffuse For Porous_Enthalpy ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
variable	“string”	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Diffuse For Porous_Enthalpy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
f	“string”	–

User_Field

Syntax

Bc Diffuse For Porous_Enthalpy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
scaling	real	1
global_var	“string”	–
time_function	“string”	–

User_Function

Syntax

Bc Diffuse For Porous_Enthalpy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc Diffuse For Porous_Enthalpy ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

User_Vector_Field

Syntax

Bc Diffuse For Porous_Enthalpy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Vector_Field ([Toggle = *toggle*] | Name = *name* | [Multiplier = *multiplier*])

Summary

User vector field enthalpy flux

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
multiplier	real	1

Porous_Species

Constant

Syntax

Bc Diffuse For Porous_Species ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | *{flux | value}* = *flux*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
flux	real	–

Exponential

Syntax

Bc Diffuse For Porous_Species ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Diffuse For Porous_Species ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
global_name	"string"	–
component	integer	0

Moving_Boundary

Syntax

Bc Diffuse For Porous_Species ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Moving_Boundary [Toggle = *toggle*]

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–

Nd_Table

Syntax

Bc Diffuse For Porous_Species ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Open_Flow

Syntax

Bc Diffuse For Porous_Species ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Open_Flow [Toggle = *toggle*]

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–

Polynomial

Syntax

Bc Diffuse For Porous_Species ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Diffuse For Porous_Species ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name.

Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
f	“string”	–

User_Field

Syntax

Bc Diffuse For Porous_Species ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc Diffuse For Porous_Species ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc Diffuse For Porous_Species ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

Solid

Constant

Syntax

Bc Diffuse For Solid ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | [Valuex = *valuex*] | [Valuey = *valuey*] | [Valuez = *valuez*])

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
valuex	real	0
valuey	real	0
valuez	real	0

Constant_Traction

Syntax

Bc Diffuse For Solid ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant_Traction ([Toggle = *toggle*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Constant traction solid flux

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
x	real	0
y	real	0
z	real	0

Electric_Traction

Syntax

Bc Diffuse For Solid ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Electric_Traction ([Toggle = *toggle*] | [Sign = *sign*])

Summary

Electric traction solid flux

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
sign	real	1

Pressure

Syntax

Bc Diffuse For Solid ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Pressure ([Toggle = *toggle*] | [User_Field_Name = *user_field_name*] | [Use_Bulk_Node = *use_bulk_node*] | [P = *p*] | [C_T = *c_t*] | [C_X = *c_x*] | [C_Y = *c_y*] | [C_Z = *c_z*] | [C_Cos_W = *c_cos_w*] | [C_Sin_W = *c_sin_w*] | [C1_Cos = *c1_cos*] | [C1_Sin = *c1_sin*])

Summary

Pressure solid flux

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
user_field_name	"string"	–
use_bulk_node	"string"	–
p	real	0
c_t	real	0
c_x	real	0
c_y	real	0
c_z	real	0
c_cos_w	real	0
c_sin_w	real	0
c1_cos	real	0
c1_sin	real	0

Pressure_User_Function

Syntax

Bc Diffuse For Solid ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Pressure_User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Pressure user function solid flux

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1

Shear_Free

Syntax

Bc Diffuse For Solid ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Shear_Free [Toggle = *toggle*]

Summary

Shear-free traction solid flux

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–

Spring_Force

Syntax

Bc Diffuse For Solid ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Spring_Force ([Toggle = *toggle*] | Mult = *Mult*)

Summary

Spring force solid flux

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Mult	real	–

Transient_Traction

Syntax

Bc Diffuse For Solid ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Transient_Traction ([Toggle = *toggle*] | [A_X = *a_x*] | [A_Y = *a_y*] | [A_Z = *a_z*] | [B_X = *b_x*] | [B_Y = *b_y*] | [B_Z = *b_z*])

Summary

Transient traction solid flux

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
a_x	real	0
a_y	real	0
a_z	real	0
b_x	real	0
b_y	real	0
b_z	real	0

Vector_String_Function

Syntax

Bc Diffuse For Solid ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Vector_String_Function ([Toggle = *toggle*] | F_X = *f_x* | F_Y = *f_y* | [F_Z = *f_z*])

Summary

Value from a string function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
f_x	“string”	–
f_y	“string”	–
f_z	“string”	–

Solid_X

Constant

Syntax

Bc Diffuse For Solid_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | Value = *value*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
value	real	–

Exponential

Syntax

Bc Diffuse For Solid_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Diffuse For Solid_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
global_name	“string”	–
component	integer	0

Nd_Table

Syntax

Bc Diffuse For Solid_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc Diffuse For Solid_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Diffuse For Solid_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f	"string"	–

User_Field

Syntax

Bc Diffuse For Solid_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc Diffuse For Solid_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc Diffuse For Solid_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

Solid_Y

Constant

Syntax

Bc Diffuse For Solid_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | Value = *value*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
value	real	–

Exponential

Syntax

Bc Diffuse For Solid_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Diffuse For Solid_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
global_name	“string”	–
component	integer	0

Nd_Table

Syntax

Bc Diffuse For Solid_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc Diffuse For Solid_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Diffuse For Solid_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f	"string"	–

User_Field

Syntax

Bc Diffuse For Solid_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
scaling	real	1
global_var	“string”	–
time_function	“string”	–

User_Function

Syntax

Bc Diffuse For Solid_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc Diffuse For Solid_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

Solid_Z

Constant

Syntax

Bc Diffuse For Solid_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | Value = *value*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
value	real	–

Exponential

Syntax

Bc Diffuse For Solid_Z ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Diffuse For Solid_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
global_name	“string”	–
component	integer	0

Nd_Table

Syntax

Bc Diffuse For Solid_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc Diffuse For Solid_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Diffuse For Solid_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f	"string"	–

User_Field

Syntax

Bc Diffuse For Solid_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc Diffuse For Solid_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc Diffuse For Solid_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
Name	“string”	–
plugin_parameters	“string”	–

Species

Constant

Syntax

Bc Diffuse For Species ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | *{flux | value}* = *flux*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
flux	real	–

Exponential

Syntax

Bc Diffuse For Species ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Free_Open_Flow

Syntax

Bc Diffuse For Species ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Free_Open_Flow [Toggle = *toggle*]

Summary

This Flux BC sets the surface normal species flux equal to the diffusive flux.

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–

Global

Syntax

Bc Diffuse For Species ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
global_name	“string”	–
component	integer	0

Interface

Syntax

Bc Diffuse For Species ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Interface [Toggle = *toggle*]

Summary

Flux BC model of species for moving interfaces

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–

Nat_Conv

Syntax

Bc Diffuse For Species ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nat_Conv ([Toggle = *toggle*] | K = *k* | Yinf = *yinf*)

Summary

natconv species flux

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
k	real	–
yinf	real	–

Nd_Table

Syntax

Bc Diffuse For Species ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Open_Adv_Flow

Syntax

Bc Diffuse For Species ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Open_Adv_Flow ([Toggle = *toggle*] | [Far_Field_Entrainment_Value = *far_field_entrainment_value*])

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
far_field_entrainment_value	real	0

Polynomial

Syntax

Bc Diffuse For Species ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Diffuse For Species ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f	"string"	–

User_Field

Syntax

Bc Diffuse For Species ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
scaling	real	1
global_var	“string”	–
time_function	“string”	–

User_Function

Syntax

Bc Diffuse For Species ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc Diffuse For Species ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

Voltage

Constant

Syntax

Bc Diffuse For Voltage ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | *{flux | value}* = *flux*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
flux	real	–

Exponential

Syntax

Bc Diffuse For Voltage ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Diffuse For Voltage ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
global_name	“string”	–
component	integer	0

Nd_Table

Syntax

Bc Diffuse For Voltage ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc Diffuse For Voltage ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Diffuse For Voltage ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f	"string"	–

User_Field

Syntax

Bc Diffuse For Voltage ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc Diffuse For Voltage ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc Diffuse For Voltage ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

Bc Diffusebp

Continuity

Constant

Syntax

Bc Diffusebp For Continuity ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant Value = *value*

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
value	real	–

Exponential

Syntax

Bc Diffusebp For Continuity ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
variable	“string”	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Diffusebp For Continuity ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global (Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Bc Diffusebp For Continuity ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc Diffusebp For Continuity ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Diffusebp For Continuity ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function F = *f*

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
f	"string"	–

User_Field

Syntax

Bc Diffusebp For Continuity ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc Diffusebp For Continuity ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Bc Diffusebp For Continuity ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin (Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
Name	“string”	–
plugin_parameters	“string”	–

Cvfem_Momentum

Capillary

Syntax

Bc Diffusebp For Cvfem_Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Capillary ()

Summary

Momentum capillary flux BP BC

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–

Capillary_Stabilization

Syntax

Bc Diffusebp For Cvfem_Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Capillary_Stabilization ([Mu0 = *mu0*] | [Multiplier = *multiplier*] | [Use_Velocity_Correction = *use_velocity_correction*])

Summary

Momentum capillary stabilization flux BP BC

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
mu0	real	0
multiplier	real	1
use_velocity_correction	integer	–

Constant

Syntax

Bc Diffusebp For Cvfem_Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Valuex = *valuex*] | [Valuey = *valuey*] | [Valuez = *valuez*])

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
valuex	real	0
valuey	real	0
valuez	real	0

Ls_Capillary

Syntax

Bc Diffusebp For Cvfem_Momentum ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Ls_Capillary ()

Summary

Momentum capillary flux BP BC using level set normal

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–

Ls_Capillary_Stabilization

Syntax

Bc Diffusebp For Cvfem_Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Ls_Capillary_Stabilization ([Mu0 = *mu0*] | [Multiplier = *multiplier*] | [Use_Velocity_Correction = *use_velocity_correction*])

Summary

Momentum capillary stabilization flux BP BC using level set normal

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
mu0	real	0
multiplier	real	1
use_velocity_correction	integer	–

Normal_Capillary_Stabilization

Syntax

Bc Diffusebp For Cvfem_Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Normal_Capillary_Stabilization ([Mu0 = *mu0*] | [Multiplier = *multiplier*] | [Use_Velocity_Correction = *use_velocity_correction*])

Summary

Momentum capillary stabilization flux BP BC

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
mu0	real	0
multiplier	real	1
use_velocity_correction	integer	–

Vector_String_Function

Syntax

Bc Diffusebp For Cvfem_Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Vector_String_Function (F_X = *f_x* | F_Y = *f_y* | [F_Z = *f_z*])

Summary

Value from a string function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
f_x	“string”	–
f_y	“string”	–
f_z	“string”	–

Cvfem_Momentum_X

Constant

Syntax

Bc Diffusebp For Cvfem_Momentum_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant Value = *value*

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
value	real	–

Exponential

Syntax

Bc Diffusebp For Cvfem_Momentum_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Diffusebp For Cvfem_Momentum_X ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global (Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Bc Diffusebp For Cvfem_Momentum_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc Diffusebp For Cvfem_Momentum_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Diffusebp For Cvfem_Momentum_X ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function F = *f*

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about

valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
f	"string"	–

User_Field

Syntax

Bc Diffusebp For Cvfem_Momentum_X ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc Diffusebp For Cvfem_Momentum_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Bc Diffusebp For Cvfem_Momentum_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin (Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
Name	“string”	–
plugin_parameters	“string”	–

Cvfem_Momentum_Y

Constant

Syntax

Bc Diffusebp For Cvfem_Momentum_Y ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant Value = *value*

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
value	real	–

Exponential

Syntax

Bc Diffusebp For Cvfem_Momentum_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
variable	“string”	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Diffusebp For Cvfem_Momentum_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global (Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Bc Diffusebp For Cvfem_Momentum_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc Diffusebp For Cvfem_Momentum_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Diffusebp For Cvfem_Momentum_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function F = *f*

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
f	"string"	–

User_Field

Syntax

Bc Diffusebp For Cvfem_Momentum_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc Diffusebp For Cvfem_Momentum_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Bc Diffusebp For Cvfem_Momentum_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin (Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
Name	“string”	–
plugin_parameters	“string”	–

Cvfem_Momentum_Z

Constant

Syntax

Bc Diffusebp For Cvfem_Momentum_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant Value = *value*

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
value	real	–

Exponential

Syntax

Bc Diffusebp For Cvfem_Momentum_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
variable	“string”	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Diffusebp For Cvfem_Momentum_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global (Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Bc Diffusebp For Cvfem_Momentum_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc Diffusebp For Cvfem_Momentum_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Diffusebp For Cvfem_Momentum_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function F = *f*

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
f	"string"	–

User_Field

Syntax

Bc Diffusebp For Cvfem_Momentum_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc Diffusebp For Cvfem_Momentum_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Bc Diffusebp For Cvfem_Momentum_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin (Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
Name	“string”	–
plugin_parameters	“string”	–

Energy

Surface_Stabilization

Syntax

Bc Diffusebp For Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Surface_Stabilization D = *d*

Summary

surface diffusion term for energy

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
d	real	–

Mesh

Capillary

Syntax

Bc Diffusebp For Mesh ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Capillary ()

Summary

Mesh capillary flux BP BC

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–

Constant

Syntax

Bc Diffusebp For Mesh ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Valuex = *valuex*] | [Valuey = *valuey*] | [Valuez = *valuez*])

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
valuex	real	0
valuey	real	0
valuez	real	0

Vector_String_Function

Syntax

Bc Diffusebp For Mesh ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Vector_String_Function (F_X = *f_x* | F_Y = *f_y* | [F_Z = *f_z*])

Summary

Value from a string function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
f_x	“string”	–
f_y	“string”	–
f_z	“string”	–

Mesh_X

Constant

Syntax

Bc Diffusebp For Mesh_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant Value = *value*

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
value	real	–

Exponential

Syntax

Bc Diffusebp For Mesh_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Diffusebp For Mesh_X ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global (Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Bc Diffusebp For Mesh_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc Diffusebp For Mesh_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Diffusebp For Mesh_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function F = *f*

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about

valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
f	"string"	–

User_Field

Syntax

Bc Diffusebp For Mesh_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc Diffusebp For Mesh_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Bc Diffusebp For Mesh_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin (Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
Name	“string”	–
plugin_parameters	“string”	–

Mesh_Y

Constant

Syntax

Bc Diffusebp For Mesh_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant Value = *value*

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
value	real	–

Exponential

Syntax

Bc Diffusebp For Mesh_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
variable	“string”	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Diffusebp For Mesh_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global (Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Bc Diffusebp For Mesh_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc Diffusebp For Mesh_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Diffusebp For Mesh_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function F = *f*

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
f	"string"	–

User_Field

Syntax

Bc Diffusebp For Mesh_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc Diffusebp For Mesh_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Bc Diffusebp For Mesh_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin (Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
Name	“string”	–
plugin_parameters	“string”	–

Mesh_Z

Constant

Syntax

Bc Diffusebp For Mesh_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant Value = *value*

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
value	real	–

Exponential

Syntax

Bc Diffusebp For Mesh_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
variable	“string”	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Diffusebp For Mesh_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global (Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Bc Diffusebp For Mesh_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc Diffusebp For Mesh_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Diffusebp For Mesh_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function F = *f*

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
f	"string"	–

User_Field

Syntax

Bc Diffusebp For Mesh_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc Diffusebp For Mesh_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Bc Diffusebp For Mesh_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin (Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
Name	“string”	–
plugin_parameters	“string”	–

Momentum

Capillary

Syntax

Bc Diffusebp For Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Capillary [Use_Node_Normal = *use_node_normal*]

Summary

Momentum capillary flux BP BC

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
use_node_normal	integer	0

Capillary_Stabilization

Syntax

Bc Diffusebp For Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Capillary_Stabilization ([Mu0 = *mu0*] | [Multiplier = *multiplier*] | [Use_Velocity_Correction = *use_velocity_correction*])

Summary

Momentum capillary stabilization flux BP BC

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
mu0	real	0
multiplier	real	1
use_velocity_correction	integer	–

Constant

Syntax

Bc Diffusebp For Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Valuex = *valuex*] | [Valuey = *valuey*] | [Valuez = *valuez*])

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
valuex	real	0
valuey	real	0
valuez	real	0

Ls_Capillary

Syntax

Bc Diffusebp For Momentum ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Ls_Capillary ()

Summary

Momentum capillary flux BP BC using level set normal

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–

Ls_Capillary_Stabilization

Syntax

Bc Diffusebp For Momentum ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Ls_Capillary_Stabilization ([Mu0 = *mu0*] | [Multiplier = *multiplier*] | [Use_Velocity_Correction = *use_velocity_correction*])

Summary

Momentum capillary stabilization flux BP BC using level set normal

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
mu0	real	0
multiplier	real	1
use_velocity_correction	integer	–

Normal_Capillary_Stabilization

Syntax

Bc Diffusebp For Momentum ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Normal_Capillary_Stabilization ([Mu0 = *mu0*] | [Multiplier = *multiplier*] | [Use_Velocity_Correction = *use_velocity_correction*])

Summary

Momentum capillary stabilization flux BP BC

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
mu0	real	0
multiplier	real	1
use_velocity_correction	integer	–

Smoothed_Capillary

Syntax

Bc Diffusebp For Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Smoothed_Capillary ()

Summary

Momentum capillary flux BP BC

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–

Vector_String_Function

Syntax

Bc Diffusebp For Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Vector_String_Function (F_X = *f_x* | F_Y = *f_y* | [F_Z = *f_z*])

Summary

Value from a string function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
f_x	"string"	–
f_y	"string"	–
f_z	"string"	–

Momentum_X

Constant

Syntax

Bc Diffusebp For Momentum_X ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant Value = *value*

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
value	real	–

Exponential

Syntax

Bc Diffusebp For Momentum_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
variable	“string”	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Diffusebp For Momentum_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global (Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Bc Diffusebp For Momentum_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc Diffusebp For Momentum_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Diffusebp For Momentum_X ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function F = *f*

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
f	"string"	–

User_Field

Syntax

Bc Diffusebp For Momentum_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc Diffusebp For Momentum_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Bc Diffusebp For Momentum_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin (Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
Name	“string”	–
plugin_parameters	“string”	–

Momentum_Y

Constant

Syntax

Bc Diffusebp For Momentum_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant Value = *value*

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
value	real	–

Exponential

Syntax

Bc Diffusebp For Momentum_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
variable	“string”	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Diffusebp For Momentum_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global (Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Bc Diffusebp For Momentum_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc Diffusebp For Momentum_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Diffusebp For Momentum_Y ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function F = *f*

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
f	"string"	–

User_Field

Syntax

Bc Diffusebp For Momentum_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc Diffusebp For Momentum_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Bc Diffusebp For Momentum_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin (Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
Name	"string"	–
plugin_parameters	"string"	–

Momentum_Z

Constant

Syntax

Bc Diffusebp For Momentum_Z ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant Value = *value*

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
value	real	–

Exponential

Syntax

Bc Diffusebp For Momentum_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
variable	“string”	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Diffusebp For Momentum_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global (Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Bc Diffusebp For Momentum_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc Diffusebp For Momentum_Z ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Diffusebp For Momentum_Z ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function F = *f*

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
f	"string"	–

User_Field

Syntax

Bc Diffusebp For Momentum_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc Diffusebp For Momentum_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Bc Diffusebp For Momentum_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin (Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
Name	“string”	–
plugin_parameters	“string”	–

Solid

Capillary

Syntax

Bc Diffusebp For Solid ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Capillary ()

Summary

Solid capillary flux BP BC

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–

Constant

Syntax

Bc Diffusebp For Solid ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Valuex = *valuex*] | [Valuey = *valuey*] | [Valuez = *valuez*])

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
valuex	real	0
valuey	real	0
valuez	real	0

Vector_String_Function

Syntax

Bc Diffusebp For Solid ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Vector_String_Function (F_X = *f_x* | F_Y = *f_y* | [F_Z = *f_z*])

Summary

Value from a string function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
f_x	"string"	–
f_y	"string"	–
f_z	"string"	–

Solid_X

Constant

Syntax

Bc Diffusebp For Solid_X ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant Value = *value*

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
value	real	–

Exponential

Syntax

Bc Diffusebp For Solid_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
variable	“string”	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Diffusebp For Solid_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global (Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Bc Diffusebp For Solid_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc Diffusebp For Solid_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Diffusebp For Solid_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function F = *f*

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
f	"string"	–

User_Field

Syntax

Bc Diffusebp For Solid_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc Diffusebp For Solid_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Bc Diffusebp For Solid_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin (Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
Name	“string”	–
plugin_parameters	“string”	–

Solid_Y

Constant

Syntax

Bc Diffusebp For Solid_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant Value = *value*

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
value	real	–

Exponential

Syntax

Bc Diffusebp For Solid_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
variable	“string”	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Diffusebp For Solid_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global (Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Bc Diffusebp For Solid_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc Diffusebp For Solid_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Diffusebp For Solid_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function F = *f*

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
f	“string”	–

User_Field

Syntax

Bc Diffusebp For Solid_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
name	“string”	–
scaling	real	1
global_var	“string”	–
time_function	“string”	–

User_Function

Syntax

Bc Diffusebp For Solid_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Bc Diffusebp For Solid_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin (Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
Name	“string”	–
plugin_parameters	“string”	–

Solid_Z

Constant

Syntax

Bc Diffusebp For Solid_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant Value = *value*

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
value	real	–

Exponential

Syntax

Bc Diffusebp For Solid_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
variable	“string”	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Diffusebp For Solid_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global (Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Bc Diffusebp For Solid_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc Diffusebp For Solid_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Diffusebp For Solid_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function F = *f*

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
f	"string"	–

User_Field

Syntax

Bc Diffusebp For Solid_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc Diffusebp For Solid_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Bc Diffusebp For Solid_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin (Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
Name	“string”	–
plugin_parameters	“string”	–

Species

Surface_Stabilization

Syntax

Bc Diffusebp For Species ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Surface_Stabilization D = *d*

Summary

surface diffusion term for species

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
d	real	–

Bc Dirichlet

Calore_User_Sub

Syntax

Bc Dirichlet For *Dof Name* ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Calore_User_Sub ([Power_Output = *power_output*] | [Flux_Output = *flux_output*] | [Toggle = *toggle*] | Name = *name* | Type = *type* | [Multiplier = *multiplier*] | [Material_Data_Block = *material_data_block*] | [Data = *data*])

Summary

Values from a Calore user subroutine

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output

If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Flux_Output

If specified, the average flux of the given BC/Src is calculated and stored into the global variable name specified as the *flux_output* argument.

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
Dof Name	string	–
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
power_output	“string”	–
flux_output	“string”	–
toggle	“string”	–
name	“string”	–
type	“string”	–
multiplier	real	1
material_data_block	“string”	–
data	“string”	–

Cantera

Syntax

Bc Dirichlet For *Dof Name* ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Cantera ([Power_Output = *power_output*] | [Flux_Output = *flux_output*] | [Toggle = *toggle*] | [Omit_Fd_Sens = *omit_fd_sens*])

Summary

Cantera enthalpy computed from temperature

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output

If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Flux_Output

If specified, the average flux of the given BC/Src is calculated and stored into the global variable name specified as the *flux_output* argument.

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Omit_Fd_Sens

Omits calculating this sensitivity when using FD sensitivities when `true`.

Parameter	Value	Default
Dof Name	string	–
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
power_output	"string"	–
flux_output	"string"	–
toggle	"string"	–
omit_fd_sens	"string"	false

Cht_Robin

Syntax

Bc Dirichlet For *Dof Name* ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Cht_Robin ([Power_Output = *power_output*] | [Flux_Output = *flux_output*] | [Toggle = *toggle*] | Temperature_Field = *temperature_field*)

Summary

Dirichlet boundary condition for interface coupled with Robin style BC CHT_ROBIN. This BC computes the heat flux for the other side.

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output

If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Flux_Output

If specified, the average flux of the given BC/Src is calculated and stored into the global variable name specified as the *flux_output* argument.

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
Dof Name	string	–
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
power_output	"string"	–
flux_output	"string"	–
toggle	"string"	–
temperature_field	"string"	–

Constant

Syntax

Bc Dirichlet For *Dof Name* ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Power_Output = *power_output*] | [Flux_Output = *flux_output*] | [Toggle = *toggle*] | Value = *value*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output

If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Flux_Output

If specified, the average flux of the given BC/Src is calculated and stored into the global variable name specified as the *flux_output* argument.

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
Dof Name	string	–
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
power_output	"string"	–
flux_output	"string"	–
toggle	"string"	–
value	real	–

Copied

Syntax

Bc Dirichlet For *Dof Name* ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Copied ([Power_Output = *power_output*] | [Flux_Output = *flux_output*] | [Toggle = *toggle*] | Source = *source*)

Summary

Copied value from another expression

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output

If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Flux_Output

If specified, the average flux of the given BC/Src is calculated and stored into the global variable name specified as the *flux_output* argument.

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
Dof Name	string	–
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
power_output	“string”	–
flux_output	“string”	–
toggle	“string”	–
source	“string”	–

Darcy

Syntax

Bc Dirichlet For *Dof Name* ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Darcy ([Gx = *gx*] | [Gy = *gy*] | [Gz = *gz*] | [Relative_To = *relative_to*])

Summary

Provides the Darcy velocity model.

Description

For phase p, the velocity provided is

$$\mathbf{u}_p = \phi S_p \mathbf{u}_{matrix} - \frac{k_{r,p}}{\mu_p} \mathbf{K}_s \left(\nabla P_p - \rho_p \mathbf{g} \right),$$

where $\mathbf{u}_{matrix}$ is the optional matrix velocity as specified in the `relative to` parameter, ϕ is the porosity, S_p is the optional phase saturation, $k_{r,p}$ is the relative permeability of the phase, μ_p is the dynamic viscosity of the phase, $\mathbf{K}_s$

is the intrinsic permeability tensor, P_p is the phase pressure, ρ_p is the density of the phase, and **g** is the gravitational acceleration vector.

If the optional phase saturation S_p is provided, the pore volume is assumed to be occupied by multiple phases (e.g., liquid / gas) in accordance to their saturations, otherwise it is occupied fully by a single phase. Note, the sum of phase saturation should equal to one i.e., individual phase volumes should sum to the total pore volume.

Additionally, if the matrix is deforming, the optional `relative to` parameter should specify which phase should be used as the matrix velocity. For instance, `relative to = solid_phase` would ensure the velocity due to the pressure gradient is relative to the deforming solid matrix.

Lastly, note the Darcy velocity is the superficial velocity and is related to the pore scale velocity through the product of porosity and optional phase saturation.

Parameter	Value	Default
Dof Name	string	–
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
gx	real	0
gy	real	0
gz	real	0
relative_to	“string”	–

Darcy_Solvent

Syntax

Bc Dirichlet For *Dof Name* ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Darcy_Solvent ([Gx = *gx*] | [Gy = *gy*] | [Gz = *gz*] | [Relative_To = *relative_to*])

Summary

This model is intended for problems where the porous species equation is being solved for pressure with a solvent species concentration as the mass term in order to solve for the velocity of that solvent species.

Parameter	Value	Default
Dof Name	string	–
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
gx	real	0
gy	real	0
gz	real	0
relative_to	"string"	–

Dirichlet_From_Temperature

Syntax

Bc Dirichlet For *Dof Name* ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Dirichlet_From_Temperature ([Temperature = *temperature*] | [Pressure = *pressure*] | [Mass_Fraction_0 = *mass_fraction_0*] | [Mass_Fraction_1 = *mass_fraction_1*] | [Mass_Fraction_2 = *mass_fraction_2*] | [Mass_Fraction_3 = *mass_fraction_3*] | [Mass_Fraction_4 = *mass_fraction_4*] | [Mass_Fraction_5 = *mass_fraction_5*] | [Mass_Fraction_6 = *mass_fraction_6*] | [Mass_Fraction_7 = *mass_fraction_7*] | [Mass_Fraction_8 = *mass_fraction_8*])

Summary

Enthalpy computed from mass fraction, temperature, and pressure

Parameter	Value	Default
Dof Name	string	–
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
temperature	real	0
pressure	real	0
mass_fraction_0	real	0
mass_fraction_1	real	0
mass_fraction_2	real	0
mass_fraction_3	real	0
mass_fraction_4	real	0
mass_fraction_5	real	0
mass_fraction_6	real	0
mass_fraction_7	real	0
mass_fraction_8	real	0

Encore_Saturation

Syntax

Bc Dirichlet For *Dof Name* ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Encore_Saturation (Name = *name* | [Result_Name = *result_name*] | [Eval_Type = *eval_type*])

Summary

A Van Genuchten model for air/water capillary pressure based on an encore function for saturation

Parameter	Value	Default
Dof Name	string	–
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
name	"string"	–
result_name	"string"	–
eval_type	"string"	–

Exponential

Syntax

Bc Dirichlet For *Dof Name* ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Power_Output = *power_output*] | [Flux_Output = *flux_output*] | [Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output

If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Flux_Output

If specified, the average flux of the given BC/Src is calculated and stored into the global variable name specified as the *flux_output* argument.

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
Dof Name	string	–
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
power_output	"string"	–
flux_output	"string"	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Fortran_User_Sub

Syntax

Bc Dirichlet For *Dof Name* ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Fortran_User_Sub ([Power_Output = *power_output*] | [Flux_Output = *flux_output*] | [Toggle = *toggle*] | [Multiplier = *multiplier*] | Name = *name* | Type = *type* | [Real_Data = *real_data*] | [Int_Data = *int_data*] | [Resource_Name = *resource_name*] | [Data = *data*] | [Material_Data_Block = *material_data_block*])

Summary

Dirichlet value from a Fortran user subroutine

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output

If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Flux_Output

If specified, the average flux of the given BC/Src is calculated and stored into the global variable name specified as the *flux_output* argument.

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
Dof Name	string	–
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
power_output	"string"	–
flux_output	"string"	–
toggle	"string"	–
multiplier	real	1
name	"string"	–
type	"string"	–
real_data	"string"	–
int_data	"string"	–
resource_name	"string"	–
data	"string"	–
material_data_block	"string"	–

From_Chemeq

Syntax

Bc Dirichlet For *Dof Name* ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = From_Chemeq ()

Summary

Species fraction of one species extracted from ChemEq

Parameter	Value	Default
Dof Name	string	–
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–

From_Mass_Fraction

Syntax

Bc Dirichlet For *Dof Name* ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = From_Mass_Fraction ()

Summary

Species fraction from mass fraction

Parameter	Value	Default
Dof Name	string	–
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–

From_Mesh_Displacement

Syntax

Bc Dirichlet For *Dof Name* ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = From_Mesh_Displacement ()

Parameter	Value	Default
Dof Name	string	–
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–

From_No_Material_Phase

Syntax

Bc Dirichlet For *Dof Name* ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = From_No_Material_Phase ([Power_Output = *power_output*] | [Flux_Output = *flux_output*] | [Toggle = *toggle*])

Summary

Copy the temperature from NO_MATERIAL_PHASE to the desired material phase.

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output

If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Flux_Output

If specified, the average flux of the given BC/Src is calculated and stored into the global variable name specified as the *flux_output* argument.

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
Dof Name	string	–
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
power_output	“string”	–
flux_output	“string”	–
toggle	“string”	–

From_Other_Phase

Syntax

Bc Dirichlet For *Dof Name* ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = From_Other_Phase ()

Summary

Get the saturation from one minus the other phase saturation

Parameter	Value	Default
Dof Name	string	–
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–

From_Species

Syntax

Bc Dirichlet For *Dof Name* ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = From_Species ()

Summary

Species fraction of one species from it's concentration and the overall concentration

Parameter	Value	Default
Dof Name	string	–
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–

From_Temperature

Syntax

Bc Dirichlet For *Dof Name* ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = From_Temperature (*{t | value}* = *t* | [Poffset = *poffset*])

Summary

Cantera enthalpy computed from temperature

Parameter	Value	Default
Dof Name	string	–
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
t	real	–
poffset	real	0

Global

Syntax

Bc Dirichlet For *Dof Name* ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Power_Output = *power_output*] | [Flux_Output = *flux_output*] | [Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output

If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Flux_Output

If specified, the average flux of the given BC/Src is calculated and stored into the global variable name specified as the *flux_output* argument.

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
Dof Name	string	–
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
power_output	"string"	–
flux_output	"string"	–
toggle	"string"	–
global_name	"string"	–
component	integer	0

Mass_Average

Syntax

Bc Dirichlet For *Dof Name* ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Mass_Average ()

Summary

Enthalpy computed from mass average of species enthalpies

Description

Defines a mixture enthalpy from mass fractions as

$$h = \frac{\sum_i Y_i h_i}{\sum_i Y_i}$$

Parameter	Value	Default
Dof Name	string	–
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–

Melting_Capillary_Darcy

Syntax

Bc Dirichlet For *Dof Name* ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Melting_Capillary_Darcy ([Melt_Dt = *melt_DT*] | [Melt_Time = *melt_time*] | [Gx = *gx*] | [Gy = *gy*] | [Gz = *gz*])

Parameter	Value	Default
Dof Name	string	–
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
melt_DT	real	5
melt_time	real	1.798e+308
gx	real	0
gy	real	0
gz	real	0

Nd_Table

Syntax

Bc Dirichlet For *Dof Name* ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Power_Output = *power_output*] | [Flux_Output = *flux_output*] | [Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output

If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Flux_Output

If specified, the average flux of the given BC/Src is calculated and stored into the global variable name specified as the *flux_output* argument.

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
Dof Name	string	–
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
power_output	“string”	–
flux_output	“string”	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc Dirichlet For *Dof Name* ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Power_Output = *power_output*] | [Flux_Output = *flux_output*] | [Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output

If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Flux_Output

If specified, the average flux of the given BC/Src is calculated and stored into the global variable name specified as the *flux_output* argument.

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
Dof Name	string	–
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
power_output	"string"	–
flux_output	"string"	–
toggle	"string"	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Dirichlet For *Dof Name* ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Power_Output = *power_output*] | [Flux_Output = *flux_output*] | [Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output

If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Flux_Output

If specified, the average flux of the given BC/Src is calculated and stored into the global variable name specified as the *flux_output* argument.

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
Dof Name	string	–
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
power_output	"string"	–
flux_output	"string"	–
toggle	"string"	–
f	"string"	–

Schloegl

Syntax

Bc Dirichlet For *Dof Name* ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Schloegl (K_Phi = *k_phi* | F = *f* | [Gx = *gx*] | [Gy = *gy*] | [Gz = *gz*] | [Proton_Name = *proton_name*])

Parameter	Value	Default
Dof Name	string	–
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
k_phi	real	–
f	real	–
gx	real	0
gy	real	0
gz	real	0
proton_name	"string"	Hp

User_Field

Syntax

Bc Dirichlet For *Dof Name* ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Power_Output = *power_output*] | [Flux_Output = *flux_output*] | [Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output

If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Flux_Output

If specified, the average flux of the given BC/Src is calculated and stored into the global variable name specified as the *flux_output* argument.

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
Dof Name	string	–
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
power_output	"string"	–
flux_output	"string"	–
toggle	"string"	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc Dirichlet For *Dof Name* ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Power_Output = *power_output*] | [Flux_Output = *flux_output*] | [Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output

If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Flux_Output

If specified, the average flux of the given BC/Src is calculated and

stored into the global variable name specified as the *flux_output* argument.

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
Dof Name	string	–
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
power_output	"string"	–
flux_output	"string"	–
toggle	"string"	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc Dirichlet For *Dof Name* ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Power_Output = *power_output*] | [Flux_Output = *flux_output*] | [Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output

If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Flux_Output

If specified, the average flux of the given BC/Src is calculated and stored into the global variable name specified as the *flux_output* argument.

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
Dof Name	string	–
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
power_output	“string”	–
flux_output	“string”	–
toggle	“string”	–
Name	“string”	–
plugin_parameters	“string”	–

Van_Genuchten_Bc

Syntax

Bc Dirichlet For *Dof Name* ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Van_Genuchten_Bc Wetting_Saturation_Expression_Name = *wetting_saturation_expression_name*

Summary

A Van Genuchten model for air/water capillary pressure based on an arbitrary function for wetting phase saturation

Parameter	Value	Default
Dof Name	string	–
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
wetting_saturation_expression_name	"string"	–

Vector_String_Function

Syntax

Bc Dirichlet For *Dof Name* ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Vector_String_Function (F_X = *f_x* | F_Y = *f_y* | [F_Z = *f_z*])

Summary

Value from a string function

Parameter	Value	Default
Dof Name	string	–
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
f_x	"string"	–
f_y	"string"	–
f_z	"string"	–

Wall_Function

Syntax

Bc Dirichlet For *Dof Name* ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Wall_Function Wall_Friction_Factor = *wall_friction_factor*

Summary

Turbulent kinetic energy using wall function

Parameter	Value	Default
Dof Name	string	–
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
wall_friction_factor	real	–

Bc Disting

Current

Constant

Syntax

Bc Disting For Current ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant Value = *value*

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
value	real	–

Exponential

Syntax

Bc Disting For Current ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
variable	“string”	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Disting For Current ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global (Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
global_name	“string”	–
component	integer	0

Nd_Table

Syntax

Bc Disting For Current ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc Disting For Current ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Disting For Current ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function F = *f*

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
f	"string"	–

User_Field

Syntax

Bc Disting For Current ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc Disting For Current ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Bc Disting For Current ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin (Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
Name	“string”	–
plugin_parameters	“string”	–

Cvfem_Energy

Constant

Syntax

Bc Disting For Cvfem_Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant Value = *value*

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
value	real	–

Exponential

Syntax

Bc Disting For Cvfem_Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
variable	“string”	–
constant	real	0
multiplier	real	1
exponent	real	–

From_Cht_Temperature

Syntax

Bc Disting For Cvfem_Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = From_Cht_Temperature ([Twall_Field = *Twall_Field*] | [Poffset = *poffset*])

Summary

CVFEM energy distinguishing condition from CHT temperature using Cantera

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
Twall_Field	"string"	nodal_cht_Tw
poffset	real	0

From_Cht_Temperature_Cpt

Syntax

Bc Disting For Cvfem_Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = From_Cht_Temperature_Cpt [Twall_Field = *Twall_Field*]

Summary

CVFEM energy distinguishing condition from CHT temperature using Cp*T model

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
Twall_Field	"string"	nodal_cht_Tw

From_Temperature

Syntax

Bc Disting For Cvfem_Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = From_Temperature (*{t | value}* = *t* | [Poffset = *poffset*])

Summary

From temperature distinguishing condition for CVFEM energy equation

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
t	real	–
poffset	real	0

Global

Syntax

Bc Disting For Cvfem_Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global (Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Bc Disting For Cvfem_Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc Disting For Cvfem_Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Disting For Cvfem_Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function F = *f*

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
f	"string"	–

User_Field

Syntax

Bc Disting For Cvfem_Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc Disting For Cvfem_Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Bc Disting For Cvfem_Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin (Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
Name	“string”	–
plugin_parameters	“string”	–

Cvfem_Level_Set

Constant

Syntax

Bc Disting For Cvfem_Level_Set ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant Value = *value*

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
value	real	–

Exponential

Syntax

Bc Disting For Cvfem_Level_Set ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
variable	“string”	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Disting For Cvfem_Level_Set ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global (Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Bc Disting For Cvfem_Level_Set ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc Disting For Cvfem_Level_Set ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Disting For Cvfem_Level_Set ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function F = *f*

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
f	"string"	–

User_Field

Syntax

Bc Disting For Cvfem_Level_Set ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc Disting For Cvfem_Level_Set ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Bc Disting For Cvfem_Level_Set ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin (Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
Name	“string”	–
plugin_parameters	“string”	–

Cvfem_Momentum

No_Slip

Syntax

Bc Disting For Cvfem_Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = No_Slip ()

Summary

No slip model for momentum distinguishing condition

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–

Porous_Robin_Interface

Syntax

Bc Disting For Cvfem_Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Porous_Robin_Interface ([Alpha = *alpha*] | Permeability_Field = *permeability_field*)

Summary

No slip model for momentum distinguishing condition

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
alpha	real	0.1
permeability_field	“string”	–

Porous_Robin_One_Region

Syntax

Bc Disting For Cvfem_Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Porous_Robin_One_Region [Alpha = *alpha*]

Summary

No slip model for momentum distinguishing condition

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
alpha	real	0.1

Cvfem_Specific_Dissipation_Rate

Constant

Syntax

Bc Disting For Cvfem_Specific_Dissipation_Rate ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant Value = *value*

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
value	real	–

Exponential

Syntax

Bc Disting For Cvfem_Specific_Dissipation_Rate ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
variable	“string”	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Disting For Cvfem_Specific_Dissipation_Rate ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global (Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
global_name	"string"	–
component	integer	0

Low_Reynolds

Syntax

Bc Disting For Cvfem_Specific_Dissipation_Rate ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Low_Reynolds ()

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–

Nd_Table

Syntax

Bc Disting For Cvfem_Specific_Dissipation_Rate ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc Disting For Cvfem_Specific_Dissipation_Rate ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Disting For Cvfem_Specific_Dissipation_Rate ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function F = *f*

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
f	"string"	–

User_Field

Syntax

Bc Disting For Cvfem_Specific_Dissipation_Rate ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc Disting For Cvfem_Specific_Dissipation_Rate ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Bc Disting For Cvfem_Specific_Dissipation_Rate ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin (Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
Name	"string"	–
plugin_parameters	"string"	–

Wall_Function

Syntax

Bc Disting For Cvfem_Specific_Dissipation_Rate ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Wall_Function ([Wall_Friction_Factor = *Wall_Friction_Factor*] | [Wall_Velocity_X = *Wall_Velocity_X*] | [Wall_Velocity_Y = *Wall_Velocity_Y*] | [Wall_Velocity_Z = *Wall_Velocity_Z*])

Summary

CVFEM specific dissipation rate distinguishing condition

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
Wall_Friction_Factor	real	9.8
Wall_Velocity_X	real	0
Wall_Velocity_Y	real	0
Wall_Velocity_Z	real	0

Cvfem_Turbulence_Dissipation_Rate

Wall_Function

Syntax

Bc Disting For Cvfem_Turbulence_Dissipation_Rate ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Wall_Function ([Wall_Friction_Factor = *Wall_Friction_Factor*] | [Wall_Velocity_X = *Wall_Velocity_X*] | [Wall_Velocity_Y = *Wall_Velocity_Y*] | [Wall_Velocity_Z = *Wall_Velocity_Z*])

Summary

CVFEM wall function turbulence dissipation rate distinguishing condition

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
Wall_Friction_Factor	real	9.8
Wall_Velocity_X	real	0
Wall_Velocity_Y	real	0
Wall_Velocity_Z	real	0

Cvfem_Turbulent_Kinetic_Energy

Wall_Function

Syntax

Bc Disting For Cvfem_Turbulent_Kinetic_Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Wall_Function ([Wall_Friction_Factor = *Wall_Friction_Factor*] | [Wall_Velocity_X = *Wall_Velocity_X*] | [Wall_Velocity_Y = *Wall_Velocity_Y*] | [Wall_Velocity_Z = *Wall_Velocity_Z*])

Summary

CVFEM wall function turbulent kinetic energy distinguishing condition

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
Wall_Friction_Factor	real	9.8
Wall_Velocity_X	real	0
Wall_Velocity_Y	real	0
Wall_Velocity_Z	real	0

Energy

Constant

Syntax

Bc Disting For Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant Value = *value*

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
value	real	–

Exponential

Syntax

Bc Disting For Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
variable	“string”	–
constant	real	0
multiplier	real	1
exponent	real	–

From_Cht_Temperature

Syntax

Bc Disting For Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = From_Cht_Temperature (Interface_Name = *Interface_Name* | Twall_Field = *Twall_Field* | Tref_Field = *Tref_Field* | H_Field = *H_Field*)

Summary

Energy distinguishing condition from conjugate heat transfer temperature

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
Interface_Name	“string”	–
Twall_Field	“string”	–
Tref_Field	“string”	–
H_Field	“string”	–

Global

Syntax

Bc Disting For Energy ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global (Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
global_name	“string”	–
component	integer	0

Nd_Table

Syntax

Bc Disting For Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc Disting For Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Disting For Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function F = *f*

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about

valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
f	"string"	–

User_Field

Syntax

Bc Disting For Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc Disting For Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Bc Disting For Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin (Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
Name	“string”	–
plugin_parameters	“string”	–

Enthalpy

Constant

Syntax

Bc Disting For Enthalpy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant Value = *value*

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
value	real	–

Exponential

Syntax

Bc Disting For Enthalpy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
variable	“string”	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Disting For Enthalpy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global (Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Bc Disting For Enthalpy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc Disting For Enthalpy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Disting For Enthalpy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function F = *f*

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
f	"string"	–

User_Field

Syntax

Bc Disting For Enthalpy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc Disting For Enthalpy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Bc Disting For Enthalpy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin (Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
Name	“string”	–
plugin_parameters	“string”	–

Extension_Speed

Advection

Syntax

Bc Disting For Extension_Speed ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Advection ()

Summary

This model has been deprecated and removed from the code.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–

Constant

Syntax

Bc Disting For Extension_Speed ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant Value = *value*

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
value	real	–

Exponential

Syntax

Bc Disting For Extension_Speed ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Disting For Extension_Speed ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global (Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
global_name	"string"	–
component	integer	0

Lens_Deposition

Syntax

Bc Disting For Extension_Speed ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Lens_Deposition (Src_X = *src_x* | Dir_X = *dir_x* | Vel_X = *vel_x* | Src_Y = *src_y* | Dir_Y = *dir_y* | Vel_Y = *vel_y* | [Src_Z = *src_z*] | [Dir_Z = *dir_z*] | [Vel_Z = *vel_z*] | Massflowrate = *massFlowRate* | T_Liq = *T_liq* | T_Solid = *T_solid* | [T_On = *t_on*] | [T_Off = *t_off*] | Radius = *radius* | Particledensity = *particleDensity*)

Summary

LENS Deposition mass source model for extension speed distinguishing condition

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
src_x	real	–
dir_x	real	–
vel_x	real	–
src_y	real	–
dir_y	real	–
vel_y	real	–
src_z	real	0
dir_z	real	0
vel_z	real	0
massFlowRate	real	–
T_liq	real	–
T_solid	real	–
t_on	real	0
t_off	real	1.798e+308
radius	real	–
particleDensity	real	–

Melting

Syntax

Bc Disting For Extension_Speed ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Melting ()

Summary

Melting model for extension speed distinguishing condition

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–

Nd_Table

Syntax

Bc Disting For Extension_Speed ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc Disting For Extension_Speed ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Pyrolysis

Syntax

Bc Disting For Extension_Speed ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Pyrolysis ([T_Ref = *t_ref*] | K = *k* | K_0 = *k_0* | N = *n* | [Temperature = *temperature*])

Summary

Pyrolysis model for extension speed distinguishing condition

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
t_ref	real	0
k	real	–
k_0	real	–
n	real	–
temperature	"string"	–

Recession

Syntax

Bc Disting For Extension_Speed ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Recession ([P_Ref = *p_ref*] | K = *k* | K_0 = *k_0* | N = *n* | [Pressure = *pressure*])

Summary

Advection model for extension speed distinguishing condition

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
p_ref	real	0
k	real	–
k_0	real	–
n	real	–
pressure	"string"	–

Scalar_String_Function

Syntax

Bc Disting For Extension_Speed ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function F = *f*

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
f	“string”	–

Solidification

Syntax

Bc Disting For Extension_Speed ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Solidification ()

Summary

Solidification model for extension speed distinguishing condition

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–

User_Field

Syntax

Bc Disting For Extension_Speed ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
name	“string”	–
scaling	real	1
global_var	“string”	–
time_function	“string”	–

User_Function

Syntax

Bc Disting For Extension_Speed ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Bc Disting For Extension_Speed ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin (Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
Name	"string"	–
plugin_parameters	"string"	–

Level_Set

Advection

Syntax

Bc Disting For Level_Set ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Advection ()

Summary

This model has been deprecated and removed from the code.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–

Constant

Syntax

Bc Disting For Level_Set ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant Value = *value*

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
value	real	–

Exponential

Syntax

Bc Disting For Level_Set ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Disting For Level_Set ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global (Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Bc Disting For Level_Set ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Outflow

Syntax

Bc Disting For Level_Set ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Outflow [V0 = *v0*]

Summary

Outflow model for level set distinguishing condition

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
v0	real	0

Polynomial

Syntax

Bc Disting For Level_Set ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Recession

Syntax

Bc Disting For Level_Set ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Recession ([P_Ref = *p_ref*] | K = *k* | K_0 = *k_0* | N = *n* | [Pressure = *pressure*])

Summary

This model has been deprecated and removed from the code.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
p_ref	real	0
k	real	–
k_0	real	–
n	real	–
pressure	"string"	–

Scalar_String_Function

Syntax

Bc Disting For Level_Set ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function F = *f*

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
f	"string"	–

User_Field

Syntax

Bc Disting For Level_Set ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
name	“string”	–
scaling	real	1
global_var	“string”	–
time_function	“string”	–

User_Function

Syntax

Bc Disting For Level_Set ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Bc Disting For Level_Set ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin (Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
Name	“string”	–
plugin_parameters	“string”	–

Lubrication

Constant

Syntax

Bc Disting For Lubrication ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant Value = *value*

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
value	real	–

Exponential

Syntax

Bc Disting For Lubrication ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Disting For Lubrication ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global (Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Bc Disting For Lubrication ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc Disting For Lubrication ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Disting For Lubrication ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function F = *f*

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about

valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
f	"string"	–

User_Field

Syntax

Bc Disting For Lubrication ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc Disting For Lubrication ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Bc Disting For Lubrication ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin (Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
Name	"string"	–
plugin_parameters	"string"	–

Mass_Balance

Constant

Syntax

Bc Disting For Mass_Balance ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant Value = *value*

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
value	real	–

Exponential

Syntax

Bc Disting For Mass_Balance ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
variable	“string”	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Disting For Mass_Balance ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global (Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Bc Disting For Mass_Balance ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc Disting For Mass_Balance ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Disting For Mass_Balance ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function F = *f*

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
f	"string"	–

User_Field

Syntax

Bc Disting For Mass_Balance ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc Disting For Mass_Balance ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Bc Disting For Mass_Balance ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin (Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
Name	“string”	–
plugin_parameters	“string”	–

Mesh

Constant

Syntax

Bc Disting For Mesh ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Valuex = *valuex*] | [Valuey = *valuey*] | [Valuez = *valuez*])

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
valuex	real	0
valuey	real	0
valuez	real	0

No_Slip

Syntax

Bc Disting For Mesh ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = No_Slip [V0 = *v0*]

Summary

No slip model for mesh distinguishing condition

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
v0	real	0

Vector_String_Function

Syntax

Bc Disting For Mesh ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Vector_String_Function (F_X = *f_x* | F_Y = *f_y* | [F_Z = *f_z*])

Summary

Value from a string function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
f_x	"string"	–
f_y	"string"	–
f_z	"string"	–

Mesh_X

Constant

Syntax

Bc Disting For Mesh_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant Value = *value*

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
value	real	–

Exponential

Syntax

Bc Disting For Mesh_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
variable	“string”	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Disting For Mesh_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global (Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
global_name	"string"	–
component	integer	0

Kinematic

Syntax

Bc Disting For Mesh_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Kinematic ()

Summary

Kinematic model for distinguishing condition has been removed, email sierra-help@sandia.gov for help.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–

Nd_Table

Syntax

Bc Disting For Mesh_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc Disting For Mesh_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Pyrolysis

Syntax

Bc Disting For Mesh_X ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Pyrolysis ()

Summary

Pyrolysis model for distinguishing condition has been removed, email sierra-help@sandia.gov for help.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–

Recession

Syntax

Bc Disting For Mesh_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Recession ()

Summary

Recession model for distinguishing condition has been removed, email sierra-help@sandia.gov for help.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–

Scalar_String_Function

Syntax

Bc Disting For Mesh_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function F = *f*

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
f	"string"	–

User_Field

Syntax

Bc Disting For Mesh_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc Disting For Mesh_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Bc Disting For Mesh_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin (Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
Name	“string”	–
plugin_parameters	“string”	–

Mesh_Y

Constant

Syntax

Bc Disting For Mesh_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant Value = *value*

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
value	real	–

Exponential

Syntax

Bc Disting For Mesh_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
variable	“string”	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Disting For Mesh_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global (Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
global_name	"string"	–
component	integer	0

Kinematic

Syntax

Bc Disting For Mesh_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Kinematic ()

Summary

Kinematic model for distinguishing condition has been removed, email sierra-help@sandia.gov for help.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–

Nd_Table

Syntax

Bc Disting For Mesh_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc Disting For Mesh_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Pyrolysis

Syntax

Bc Disting For Mesh_Y ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Pyrolysis ()

Summary

Pyrolysis model for distinguishing condition has been removed, email sierra-help@sandia.gov for help.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–

Recession

Syntax

Bc Disting For Mesh_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Recession ()

Summary

Recession model for distinguishing condition has been removed, email sierra-help@sandia.gov for help.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–

Scalar_String_Function

Syntax

Bc Disting For Mesh_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function F = *f*

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
f	“string”	–

User_Field

Syntax

Bc Disting For Mesh_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
name	“string”	–
scaling	real	1
global_var	“string”	–
time_function	“string”	–

User_Function

Syntax

Bc Disting For Mesh_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Bc Disting For Mesh_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin (Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
Name	“string”	–
plugin_parameters	“string”	–

Mesh_Z

Constant

Syntax

Bc Disting For Mesh_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant Value = *value*

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
value	real	–

Exponential

Syntax

Bc Disting For Mesh_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
variable	“string”	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Disting For Mesh_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global (Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
global_name	"string"	–
component	integer	0

Kinematic

Syntax

Bc Disting For Mesh_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Kinematic ()

Summary

Kinematic model for distinguishing condition has been removed, email sierra-help@sandia.gov for help.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–

Nd_Table

Syntax

Bc Disting For Mesh_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc Disting For Mesh_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Pyrolysis

Syntax

Bc Disting For Mesh_Z ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Pyrolysis ()

Summary

Pyrolysis model for distinguishing condition has been removed, email sierra-help@sandia.gov for help.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–

Recession

Syntax

Bc Disting For Mesh_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Recession ()

Summary

Recession model for distinguishing condition has been removed, email sierra-help@sandia.gov for help.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–

Scalar_String_Function

Syntax

Bc Disting For Mesh_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function F = *f*

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
f	"string"	–

User_Field

Syntax

Bc Disting For Mesh_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc Disting For Mesh_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Bc Disting For Mesh_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin (Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
Name	“string”	–
plugin_parameters	“string”	–

Momentum

Constant

Syntax

Bc Disting For Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Valuex = *valuex*] | [Valuey = *valuey*] | [Valuez = *valuez*])

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
valuex	real	0
valuey	real	0
valuez	real	0

Darcy_Leak

Syntax

Bc Disting For Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Darcy_Leak ([P_Ref = *p_ref*] | [L = *l*] | K = *k* | [Pressure = *pressure*])

Summary

Darcy leak model for momentum distinguishing condition

Description

This model implements a vector distinguishing condition that allows fluid to "leak" through a boundary according to a Darcy's Law like relationship:

$$\boldsymbol{v} - \dot{\boldsymbol{x}} \cdot \boldsymbol{n}\boldsymbol{n} - \frac{K}{L\mu}\left(P - P_{ref}\right)\boldsymbol{n} = \boldsymbol{0}.$$

Here $\boldsymbol{n}$ is the outward unit normal to the boundary, $\boldsymbol{v}$ is the velocity, $\dot{\boldsymbol{x}}$ is the time derivative of the mesh boundary position and P is the fluid pressure. K is the permeability of the boundary, μ is the viscosity, P_{ref} is a reference pressure (defaults to zero) and L is a length scale for the pressure drop (defaults to unity).

Note that the second term ensures that the resulting velocity is relative to any boundary motion that arises in moving mesh problems. For problems on fixed meshes this term is automatically excluded.

An alternate pressure field can be used by provided the optional parameter `PRESSURE`.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
p_ref	real	0
l	real	1
k	real	–
pressure	"string"	–

Electroosmotic_Velocity

Syntax

Bc Disting For Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Electroosmotic_Velocity C = *c*

Summary

Electro-osmotic velocity model for momentum distinguishing condition

Description

This distinguishing condition sets the velocity to simply be a constant multiple of the electric field,

$$\boldsymbol{v} - c\boldsymbol{E} = \boldsymbol{0}$$

where $\boldsymbol{v}$ is the fluid velocity, $\boldsymbol{E}$ is the electric field and c is the constant C supplied by the user.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
c	real	–

No_Slip

Syntax

Bc Disting For Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = No_Slip [V0 = *v0*]

Summary

No slip model for momentum distinguishing condition

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
v0	real	0

Vector_String_Function

Syntax

Bc Disting For Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Vector_String_Function (F_X = *f_x* | F_Y = *f_y* | [F_Z = *f_z*])

Summary

Value from a string function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
f_x	“string”	–
f_y	“string”	–
f_z	“string”	–

Vector_User_Function_Disting

Syntax

Bc Disting For Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Vector_User_Function_Disting ([Name_X = *name_x*] | [Name_Y = *name_y*] | [Name_Z = *name_z*] | X = *x* | [Multiplier = *multiplier*])

Summary

Vector user function model for momentum distinguishing condition

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
name_x	“string”	–
name_y	“string”	–
name_z	“string”	–
x	“string”	–
multiplier	real	1

Wetting_Speed_Blake_Ls

Syntax

Bc Disting For Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Wetting_Speed_Blake_Ls (V_W = *v_w* | G = *g* | Theta = *theta* | Width = *width* | [Tau = *tau*])

Summary

Blake level set wetting speed model for momentum distinguishing condition

Description

The wetting condition is governed by

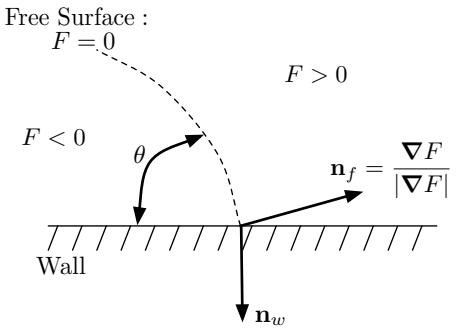
$$\boldsymbol{v} - \delta(F) v_w \sinh\left(g\left(\cos\theta_s - \cos\theta\right)\right)\boldsymbol{t}_w + \delta(F)\tau\dot{\boldsymbol{v}} = 0$$

where $\boldsymbol{v}$ is the fluid velocity and $\boldsymbol{t}_w$ is the tangent to the wall. The function $\delta(F)$, where F is the level set distance field, is given by

$$\delta(F) = \frac{1}{2}\left(1 + \cos\frac{\pi F}{\alpha}\right)$$

when $|F| < \alpha$ and zero elsewhere. Here, α is the half of the `WIDTH` parameter. The term involving τ is a transient relaxation term. By default, $\tau = 0$.

This distinguishing condition is a function of the static and observed contact angles. The static contact angle θ_s, supplied by the `THETA` parameter, is fixed. The observed contact angle θ is computed from the current state of the solution as illustrated in the following diagram. The important point here is that the contact angle is measured through the negative side of the distance function which is denoted `PHASE_A` in Aria.



Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
v_w	real	–
g	real	–
theta	real	–
width	real	–
tau	real	0

Momentum_X

Constant

Syntax

Bc Disting For Momentum_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant Value = *value*

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
value	real	–

Exponential

Syntax

Bc Disting For Momentum_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Disting For Momentum_X ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global (Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Bc Disting For Momentum_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc Disting For Momentum_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Disting For Momentum_X ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function F = *f*

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about

valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
f	"string"	–

User_Field

Syntax

Bc Disting For Momentum_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc Disting For Momentum_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Bc Disting For Momentum_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin (Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
Name	"string"	–
plugin_parameters	"string"	–

Momentum_Y

Constant

Syntax

Bc Disting For Momentum_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant Value = *value*

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
value	real	–

Exponential

Syntax

Bc Disting For Momentum_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
variable	“string”	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Disting For Momentum_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global (Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Bc Disting For Momentum_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc Disting For Momentum_Y ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Disting For Momentum_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function F = *f*

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
f	“string”	–

User_Field

Syntax

Bc Disting For Momentum_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
name	“string”	–
scaling	real	1
global_var	“string”	–
time_function	“string”	–

User_Function

Syntax

Bc Disting For Momentum_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Bc Disting For Momentum_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin (Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
Name	"string"	–
plugin_parameters	"string"	–

Momentum_Z

Constant

Syntax

Bc Disting For Momentum_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant Value = *value*

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
value	real	–

Exponential

Syntax

Bc Disting For Momentum_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
variable	“string”	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Disting For Momentum_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global (Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Bc Disting For Momentum_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc Disting For Momentum_Z ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Disting For Momentum_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function F = *f*

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
f	“string”	–

User_Field

Syntax

Bc Disting For Momentum_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
name	“string”	–
scaling	real	1
global_var	“string”	–
time_function	“string”	–

User_Function

Syntax

Bc Disting For Momentum_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Bc Disting For Momentum_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin (Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
Name	"string"	–
plugin_parameters	"string"	–

Porous_Enthalpy

Constant

Syntax

Bc Disting For Porous_Enthalpy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant Value = *value*

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
value	real	–

Exponential

Syntax

Bc Disting For Porous_Enthalpy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
variable	“string”	–
constant	real	0
multiplier	real	1
exponent	real	–

From_Cht_Temperature

Syntax

Bc Disting For Porous_Enthalpy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = From_Cht_Temperature (Interface_Name = *Interface_Name* | Twall_Field = *Twall_Field* | Tref_Field = *Tref_Field* | H_Field = *H_Field*)

Summary

Porous enthalpy distinguishing condition from conjugate heat transfer temperature

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
Interface_Name	“string”	–
Twall_Field	“string”	–
Tref_Field	“string”	–
H_Field	“string”	–

Global

Syntax

Bc Disting For Porous_Enthalpy ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global (Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
global_name	“string”	–
component	integer	0

Nd_Table

Syntax

Bc Disting For Porous_Enthalpy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc Disting For Porous_Enthalpy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Disting For Porous_Enthalpy ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function F = *f*

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about

valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
f	"string"	–

User_Field

Syntax

Bc Disting For Porous_Enthalpy ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc Disting For Porous_Enthalpy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Bc Disting For Porous_Enthalpy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin (Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
Name	“string”	–
plugin_parameters	“string”	–

Solid

Constant

Syntax

Bc Disting For Solid ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Valuex = *valuex*] | [Valuey = *valuey*] | [Valuez = *valuez*])

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
valuex	real	0
valuey	real	0
valuez	real	0

No_Slip

Syntax

Bc Disting For Solid ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = No_Slip [V0 = *v0*]

Summary

No slip model for SOLID distinguishing condition

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
v0	real	0

Vector_String_Function

Syntax

Bc Disting For Solid ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Vector_String_Function (F_X = *f_x* | F_Y = *f_y* | [F_Z = *f_z*])

Summary

Value from a string function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
f_x	"string"	–
f_y	"string"	–
f_z	"string"	–

Solid_X

Constant

Syntax

Bc Disting For Solid_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant Value = *value*

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
value	real	–

Exponential

Syntax

Bc Disting For Solid_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
variable	“string”	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Disting For Solid_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global (Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
global_name	"string"	–
component	integer	0

Kinematic

Syntax

Bc Disting For Solid_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Kinematic ()

Summary

Kinematic model for distinguishing condition. Will be deprecated in 4.59, email sierra-help@sandia.gov for help.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–

Nd_Table

Syntax

Bc Disting For Solid_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc Disting For Solid_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Pyrolysis

Syntax

Bc Disting For Solid_X ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Pyrolysis ()

Summary

Pyrolysis model for distinguishing condition has been removed , email sierra-help@sandia.gov for help.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–

Recession

Syntax

Bc Disting For Solid_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Recession ()

Summary

Recession model for distinguishing condition has been removed, email sierra-help@sandia.gov for help.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–

Scalar_String_Function

Syntax

Bc Disting For Solid_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function F = *f*

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
f	"string"	–

User_Field

Syntax

Bc Disting For Solid_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc Disting For Solid_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Bc Disting For Solid_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin (Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
Name	"string"	–
plugin_parameters	"string"	–

Solid_Y

Constant

Syntax

Bc Disting For Solid_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant Value = *value*

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
value	real	–

Exponential

Syntax

Bc Disting For Solid_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
variable	“string”	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Disting For Solid_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global (Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
global_name	"string"	–
component	integer	0

Kinematic

Syntax

Bc Disting For Solid_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Kinematic ()

Summary

Kinematic model for distinguishing condition. Will be deprecated in 4.59, email sierra-help@sandia.gov for help.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–

Nd_Table

Syntax

Bc Disting For Solid_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc Disting For Solid_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Pyrolysis

Syntax

Bc Disting For Solid_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Pyrolysis ()

Summary

Pyrolysis model for distinguishing condition has been removed , email sierra-help@sandia.gov for help.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–

Recession

Syntax

Bc Disting For Solid_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Recession ()

Summary

Recession model for distinguishing condition has been removed, email sierra-help@sandia.gov for help.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–

Scalar_String_Function

Syntax

Bc Disting For Solid_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function F = *f*

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
f	"string"	–

User_Field

Syntax

Bc Disting For Solid_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc Disting For Solid_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Bc Disting For Solid_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin (Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
Name	"string"	–
plugin_parameters	"string"	–

Solid_Z

Constant

Syntax

Bc Disting For Solid_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant Value = *value*

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
value	real	–

Exponential

Syntax

Bc Disting For Solid_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
variable	“string”	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Disting For Solid_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global (Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
global_name	"string"	–
component	integer	0

Kinematic

Syntax

Bc Disting For Solid_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Kinematic ()

Summary

Kinematic model for distinguishing condition. Will be deprecated in 4.59, email sierra-help@sandia.gov for help.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–

Nd_Table

Syntax

Bc Disting For Solid_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc Disting For Solid_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Pyrolysis

Syntax

Bc Disting For Solid_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Pyrolysis ()

Summary

Pyrolysis model for distinguishing condition has been removed , email sierra-help@sandia.gov for help.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–

Recession

Syntax

Bc Disting For Solid_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Recession ()

Summary

Recession model for distinguishing condition has been removed, email sierra-help@sandia.gov for help.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–

Scalar_String_Function

Syntax

Bc Disting For Solid_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function F = *f*

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
f	"string"	–

User_Field

Syntax

Bc Disting For Solid_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc Disting For Solid_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Bc Disting For Solid_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin (Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
Name	“string”	–
plugin_parameters	“string”	–

Species

Constant

Syntax

Bc Disting For Species ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant Value = *value*

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
value	real	–

Exponential

Syntax

Bc Disting For Species ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
variable	“string”	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Disting For Species ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global (Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Bc Disting For Species ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc Disting For Species ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Disting For Species ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function F = *f*

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
f	"string"	–

User_Field

Syntax

Bc Disting For Species ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc Disting For Species ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Bc Disting For Species ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin (Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
Name	“string”	–
plugin_parameters	“string”	–

Voltage

Constant

Syntax

Bc Disting For Voltage ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant Value = *value*

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
value	real	–

Exponential

Syntax

Bc Disting For Voltage ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
variable	“string”	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Disting For Voltage ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global (Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Bc Disting For Voltage ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc Disting For Voltage ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Disting For Voltage ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function F = *f*

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
f	"string"	–

User_Field

Syntax

Bc Disting For Voltage ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc Disting For Voltage ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Bc Disting For Voltage ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin (Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
Name	"string"	–
plugin_parameters	"string"	–

Bc Edge

Cvfem_Momentum

Wetting

Syntax

Bc Edge For Cvfem_Momentum ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) Where *SidePart0Name* Intersects *SidePart1Name* = Wetting Theta_S = *theta_s*

Summary

Sharp wetting edge source term

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
SidePart0Name	string	–
SidePart1Name	string	–
theta_s	real	–

Wetting_Outflow

Syntax

Bc Edge For Cvfem_Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) Where *SidePart0Name* Intersects *SidePart1Name* = Wetting_Outflow ()

Summary

Sharp wetting edge source term for outflow

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
SidePart0Name	string	–
SidePart1Name	string	–

Momentum

Wetting

Syntax

Bc Edge For Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) Where *SidePart0Name* Intersects *SidePart1Name* = Wetting Theta_S = *theta_s*

Summary

Sharp wetting edge source term

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
SidePart0Name	string	–
SidePart1Name	string	–
theta_s	real	–

Wetting_Outflow

Syntax

Bc Edge For Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) Where *SidePart0Name* Intersects *SidePart1Name* = Wetting_Outflow ()

Summary

Sharp wetting edge source term for outflow

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
SidePart0Name	string	–
SidePart1Name	string	–

Bc Flux

Ampere

Constant

Syntax

Bc Flux For Ampere ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | [Valuex = *valuex*] | [Valuey = *valuey*] | [Valuez = *valuez*])

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
valuex	real	0
valuey	real	0
valuez	real	0

Vector_String_Function

Syntax

Bc Flux For Ampere ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Vector_String_Function ([Toggle = *toggle*] | F_X = *f_x* | F_Y = *f_y* | [F_Z = *f_z*])

Summary

Value from a string function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f_x	"string"	–
f_y	"string"	–
f_z	"string"	–

Ampere_X

Constant

Syntax

Bc Flux For Ampere_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | Value = *value*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
value	real	–

Exponential

Syntax

Bc Flux For Ampere_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Flux For Ampere_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
global_name	“string”	–
component	integer	0

Nd_Table

Syntax

Bc Flux For Ampere_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc Flux For Ampere_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Flux For Ampere_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f	"string"	–

User_Field

Syntax

Bc Flux For Ampere_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc Flux For Ampere_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc Flux For Ampere_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

Ampere_Y

Constant

Syntax

Bc Flux For Ampere_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | Value = *value*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
value	real	–

Exponential

Syntax

Bc Flux For Ampere_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Flux For Ampere_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
global_name	“string”	–
component	integer	0

Nd_Table

Syntax

Bc Flux For Ampere_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc Flux For Ampere_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Flux For Ampere_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f	"string"	–

User_Field

Syntax

Bc Flux For Ampere_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc Flux For Ampere_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc Flux For Ampere_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

Ampere_Z

Constant

Syntax

Bc Flux For Ampere_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | Value = *value*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
value	real	–

Exponential

Syntax

Bc Flux For Ampere_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Flux For Ampere_Z ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
global_name	“string”	–
component	integer	0

Nd_Table

Syntax

Bc Flux For Ampere_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc Flux For Ampere_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Flux For Ampere_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f	"string"	–

User_Field

Syntax

Bc Flux For Ampere_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc Flux For Ampere_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc Flux For Ampere_Z ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

Charge_Density

Constant

Syntax

Bc Flux For Charge_Density ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | *{flux | value}* = *flux*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
flux	real	–

Exponential

Syntax

Bc Flux For Charge_Density ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Flux For Charge_Density ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
global_name	“string”	–
component	integer	0

Nat_E

Syntax

Bc Flux For Charge_Density ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nat_E ([Toggle = *toggle*] | [Sigma = *sigma*])

Summary

This model is now deprecated

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
sigma	“string”	–

Nd_Table

Syntax

Bc Flux For Charge_Density ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc Flux For Charge_Density ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
variable	“string”	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Flux For Charge_Density ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
f	“string”	–

User_Field

Syntax

Bc Flux For Charge_Density ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
scaling	real	1
global_var	“string”	–
time_function	“string”	–

User_Function

Syntax

Bc Flux For Charge_Density ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc Flux For Charge_Density ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

Continuity

Constant

Syntax

Bc Flux For Continuity ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | Value = *value*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
value	real	–

Exponential

Syntax

Bc Flux For Continuity ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Extrapolated_Pressure

Syntax

Bc Flux For Continuity ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Extrapolated_Pressure [Toggle = *toggle*]

Summary

This model has been deprecated and removed from the code.

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–

Global

Syntax

Bc Flux For Continuity ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Bc Flux For Continuity ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc Flux For Continuity ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
variable	“string”	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Pspg

Syntax

Bc Flux For Continuity ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Pspg ([Toggle = *toggle*] | [Tausurfacefactor = *TauSurfaceFactor*] | [Tausurfacejacobianfactor = *TauSurfaceJacobianFactor*])

Summary

Continuity PSPG flux BC

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
TauSurfaceFactor	real	1
TauSurfaceJacobianFactor	real	1

Scalar_String_Function

Syntax

Bc Flux For Continuity ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f	"string"	–

User_Field

Syntax

Bc Flux For Continuity ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc Flux For Continuity ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc Flux For Continuity ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
Name	“string”	–
plugin_parameters	“string”	–

Current

Calore_User_Sub

Syntax

Bc Flux For Current ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Calore_User_Sub ([Toggle = *toggle*] | Name = *name* | Type = *type* | [Multiplier = *multiplier*] | [Material_Data_Block = *material_data_block*] | [Data = *data*] | [Scaling_Field = *scaling_field*])

Summary

Values from a Calore user subroutine

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
type	“string”	–
multiplier	real	1
material_data_block	“string”	–
data	“string”	–
scaling_field	“string”	–

Constant

Syntax

Bc Flux For Current ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | *{flux | value}* = *flux*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
flux	real	–

Constant_With_Cutoff_Voltage

Syntax

Bc Flux For Current ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant_With_Cutoff_Voltage ([Toggle = *toggle*] | Value = *value* | Cutoff_Value = *cutoff_value*)

Summary

Apply a constant current until a cutoff voltage is crossed anywhere on the surface. After that apply 0 current.

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
value	real	–
cutoff_value	real	–

Exponential

Syntax

Bc Flux For Current ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
variable	“string”	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Flux For Current ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Bc Flux For Current ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc Flux For Current ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Resistive_Load

Syntax

Bc Flux For Current ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Resistive_Load ([Toggle = *toggle*] | [Area = *Area*] | [Referencevoltage = *ReferenceVoltage*])

Summary

Model a boundary attached to a resistive load. The local applied current density is V / (R A) where V is the local voltage, R is the specified resistance, and A is the area of the surface the BC is applied to.

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Area	real	-1
ReferenceVoltage	real	0

Scalar_String_Function

Syntax

Bc Flux For Current ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f	"string"	–

User_Field

Syntax

Bc Flux For Current ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc Flux For Current ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc Flux For Current ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

Cvfem_Continuity

Constant

Syntax

Bc Flux For Cvfem_Continuity ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | Value = *value*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
value	real	–

Exponential

Syntax

Bc Flux For Cvfem_Continuity ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Flux For Cvfem_Continuity ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
global_name	"string"	–
component	integer	0

Inflow

Syntax

Bc Flux For Cvfem_Continuity ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Inflow [Toggle = *toggle*]

Summary

IBP continuity inflow; nodal interp

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–

Interface

Syntax

Bc Flux For Cvfem_Continuity ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Interface [Toggle = *toggle*]

Summary

INTERFACE CVFEM continuity flux

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–

Mass_Flux

Syntax

Bc Flux For Cvfem_Continuity ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Mass_Flux ([Toggle = *toggle*] | Total_Mdot = *total_mdot*)

Summary

Mass flux CVFEM continuity flux

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
total_mdot	real	–

Nd_Table

Syntax

Bc Flux For Cvfem_Continuity ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Open_Flow

Syntax

Bc Flux For Cvfem_Continuity ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Open_Flow ([Toggle = *toggle*] | [Pressure = *pressure*] | [Total_Pressure = *total_pressure*])

Summary

Open bc for continuity

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
pressure	real	0
total_pressure	real	0

Polynomial

Syntax

Bc Flux For Cvfem_Continuity ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
variable	“string”	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Porous_Robin_Averaged_Coeff

Syntax

Bc Flux For Cvfem_Continuity ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Porous_Robin_Averaged_Coeff ([Toggle = *toggle*] | P_Field = *P_Field* | Massflux_Field = *MassFlux_Field* | Oppositecoeff_Field = *OppositeCoeff_Field*)

Summary

Boundary condition for loose coupling of continuity to a free fluid region

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
P_Field	"string"	–
MassFlux_Field	"string"	–
OppositeCoeff_Field	"string"	–

Porous_Robin_One_Region

Syntax

Bc Flux For Cvfem_Continuity ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Porous_Robin_One_Region [Toggle = *toggle*]

Summary

Boundary condition for loose coupling of continuity to a free fluid region

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–

Scalar_String_Function

Syntax

Bc Flux For Cvfem_Continuity ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
f	“string”	–

Simple_Inflow

Syntax

Bc Flux For Cvfem_Continuity ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Simple_Inflow ([Toggle = *toggle*] | *{density | value}* = *Density* | [Velocity_X = *Velocity_X*] | [Velocity_Y = *Velocity_Y*] | [Velocity_Z = *Velocity_Z*])

Summary

IBP continuity inflow; specified dens and u

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Density	real	–
Velocity_X	real	0
Velocity_Y	real	0
Velocity_Z	real	0

User_Field

Syntax

Bc Flux For Cvfem_Continuity ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc Flux For Cvfem_Continuity ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc Flux For Cvfem_Continuity ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
Name	“string”	–
plugin_parameters	“string”	–

Cvfem_Density_Edge_Gradient_Projection

Constant

Syntax

Bc Flux For Cvfem_Density_Edge_Gradient_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | Value = *value*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
value	real	–

Exponential

Syntax

Bc Flux For Cvfem_Density_Edge_Gradient_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Flux For Cvfem_Density_Edge_Gradient_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Bc Flux For Cvfem_Density_Edge_Gradient_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc Flux For Cvfem_Density_Edge_Gradient_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Flux For Cvfem_Density_Edge_Gradient_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f	"string"	–

Simple_Interp

Syntax

Bc Flux For Cvfem_Density_Edge_Gradient_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Simple_Interp [Toggle = *toggle*]

Summary

CVFEM edge-based projection simple interpolation flux BC

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–

User_Field

Syntax

Bc Flux For Cvfem_Density_Edge_Gradient_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
scaling	real	1
global_var	“string”	–
time_function	“string”	–

User_Function

Syntax

Bc Flux For Cvfem_Density_Edge_Gradient_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc Flux For Cvfem_Density_Edge_Gradient_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

Cvfem_Dispersed_Continuity

Constant

Syntax

Bc Flux For Cvfem_Dispersed_Continuity ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | Value = *value*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
value	real	–

Exponential

Syntax

Bc Flux For Cvfem_Dispersed_Continuity ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Flux For Cvfem_Dispersed_Continuity ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
global_name	"string"	–
component	integer	0

Inflow

Syntax

Bc Flux For Cvfem_Dispersed_Continuity ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Inflow [Toggle = *toggle*]

Summary

Inflow mass flux for dispersed phase continuity equation; nodal interpolation

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–

Nd_Table

Syntax

Bc Flux For Cvfem_Dispersed_Continuity ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Open_Flow

Syntax

Bc Flux For Cvfem_Dispersed_Continuity ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Open_Flow [Toggle = *toggle*]

Summary

Open bc for dispersed continuity

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–

Polynomial

Syntax

Bc Flux For Cvfem_Dispersed_Continuity ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset,

X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Flux For Cvfem_Dispersed_Continuity ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be

a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f	"string"	–

Simple_Inflow

Syntax

Bc Flux For Cvfem_Dispersed_Continuity ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Simple_Inflow ([Toggle = *toggle*] | [Velocity_X = *Velocity_X*] | [Velocity_Y = *Velocity_Y*] | [Velocity_Z = *Velocity_Z*])

Summary

Inflow mass flux for dispersed continuity equation; specified density and velocity

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Velocity_X	real	0
Velocity_Y	real	0
Velocity_Z	real	0

User_Field

Syntax

Bc Flux For Cvfem_Dispersed_Continuity ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc Flux For Cvfem_Dispersed_Continuity ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc Flux For Cvfem_Dispersed_Continuity ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

Cvfem_Dispersed_Momentum

Constant

Syntax

Bc Flux For Cvfem_Dispersed_Momentum ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | [Valuex = *valuex*] | [Valuey = *valuey*] | [Valuez = *valuez*])

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
valuex	real	0
valuey	real	0
valuez	real	0

Open_Adv_Flow

Syntax

Bc Flux For Cvfem_Dispersed_Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Open_Adv_Flow ([Toggle = *toggle*] | [Coflow_Entrainment_Valuex = *Coflow_Entrainment_ValueX*] | [Coflow_Entrainment_Valuey = *Coflow_Entrainment_ValueY*] | [Coflow_Entrainment_Valuez = *Coflow_Entrainment_ValueZ*])

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Coflow_Entrainment_ValueX	real	0
Coflow_Entrainment_ValueY	real	0
Coflow_Entrainment_ValueZ	real	0

Open_Flow

Syntax

Bc Flux For Cvfem_Dispersed_Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Open_Flow [Toggle = *toggle*]

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–

Symmetry_Flow

Syntax

Bc Flux For Cvfem_Dispersed_Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Symmetry_Flow [Toggle = *toggle*]

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–

Vector_String_Function

Syntax

Bc Flux For Cvfem_Dispersed_Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Vector_String_Function ([Toggle = *toggle*] | F_X = *f_x* | F_Y = *f_y* | [F_Z = *f_z*])

Summary

Value from a string function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
f_x	“string”	–
f_y	“string”	–
f_z	“string”	–

Cvfem_Dispersed_Momentum_X

Constant

Syntax

Bc Flux For Cvfem_Dispersed_Momentum_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | Value = *value*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
value	real	–

Exponential

Syntax

Bc Flux For Cvfem_Dispersed_Momentum_X ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Flux For Cvfem_Dispersed_Momentum_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Bc Flux For Cvfem_Dispersed_Momentum_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc Flux For Cvfem_Dispersed_Momentum_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Flux For Cvfem_Dispersed_Momentum_X ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f	"string"	–

User_Field

Syntax

Bc Flux For Cvfem_Dispersed_Momentum_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc Flux For Cvfem_Dispersed_Momentum_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc Flux For Cvfem_Dispersed_Momentum_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

Cvfem_Dispersed_Momentum_Y

Constant

Syntax

Bc Flux For Cvfem_Dispersed_Momentum_Y ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | Value = *value*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
value	real	–

Exponential

Syntax

Bc Flux For Cvfem_Dispersed_Momentum_Y ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Flux For Cvfem_Dispersed_Momentum_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Bc Flux For Cvfem_Dispersed_Momentum_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc Flux For Cvfem_Dispersed_Momentum_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Flux For Cvfem_Dispersed_Momentum_Y ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f	"string"	–

User_Field

Syntax

Bc Flux For Cvfem_Dispersed_Momentum_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc Flux For Cvfem_Dispersed_Momentum_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc Flux For Cvfem_Dispersed_Momentum_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

Cvfem_Dispersed_Momentum_Z

Constant

Syntax

Bc Flux For Cvfem_Dispersed_Momentum_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | Value = *value*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
value	real	–

Exponential

Syntax

Bc Flux For Cvfem_Dispersed_Momentum_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Flux For Cvfem_Dispersed_Momentum_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Bc Flux For Cvfem_Dispersed_Momentum_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc Flux For Cvfem_Dispersed_Momentum_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Flux For Cvfem_Dispersed_Momentum_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f	"string"	–

User_Field

Syntax

Bc Flux For Cvfem_Dispersed_Momentum_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc Flux For Cvfem_Dispersed_Momentum_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc Flux For Cvfem_Dispersed_Momentum_Z ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

Cvfem_Div_Projection

Constant

Syntax

Bc Flux For Cvfem_Div_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | [Valuexx = *valuexx*] | [Valuexy = *valuexy*] | [Valuexz = *valuexz*] | [Valueyx = *valueyx*] | [Valueyy = *valueyy*] | [Valueyz = *valueyz*] | [Valuezx = *valuezx*] | [Valuezy = *valuezy*] | [Valuezz = *valuezz*] | [Valuet11 = *valuet11*] | [Valuet12 = *valuet12*] | [Valuet13 = *valuet13*] | [Valuet21 = *valuet21*] | [Valuet22 = *valuet22*] | [Valuet23 = *valuet23*] | [Valuet31 = *valuet31*] | [Valuet32 = *valuet32*] | [Valuet33 = *valuet33*])

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
valuexx	real	0
valuexy	real	0
valuexz	real	0
valueyx	real	0
valueyy	real	0
valueyz	real	0
valuezx	real	0
valuezy	real	0
valuezz	real	0
valuet11	real	0
valuet12	real	0
valuet13	real	0
valuet21	real	0
valuet22	real	0
valuet23	real	0
valuet31	real	0
valuet32	real	0
valuet33	real	0

Simple_Interp_Tensor

Syntax

Bc Flux For Cvfem_Div_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Simple_Interp_Tensor ([Toggle = *toggle*] | Source = *source*)

Summary

CVFEM divergence projection simple interpolation tensor flux BC

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
source	"string"	–

Simple_Interp_Vector

Syntax

Bc Flux For Cvfem_Div_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Simple_Interp_Vector ([Toggle = *toggle*] | Source = *source*)

Summary

CVFEM divergence projection simple interpolation vector flux BC

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
source	"string"	–

Tensor_String_Function

Syntax

Bc Flux For Cvfem_Div_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Tensor_String_Function ([Toggle = *toggle*] | F_Xx = *f_xx* | F_Yy = *f_yy* | [F_Xy = *f_xy*] | [F_Zz = *f_zz*] | [F_Xz = *f_xz*] | [F_Yz = *f_yz*] | [F_Yx = *f_yx*] | [F_Zx = *f_zx*] | [F_Zy = *f_zy*])

Summary

Value from a string function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
f_xx	“string”	–
f_yy	“string”	–
f_xy	“string”	0.0
f_zz	“string”	–
f_xz	“string”	0.0
f_yz	“string”	0.0
f_yx	“string”	0.0
f_zx	“string”	0.0
f_zy	“string”	0.0

Tensor_String_Function_Symmetric

Syntax

Bc Flux For Cvfem_Div_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Tensor_String_Function_Symmetric ([Toggle = *toggle*] | F_Xx = *f_xx* | F_Yy = *f_yy* | [F_Xy = *f_xy*] | [F_Zz = *f_zz*] | [F_Xz = *f_xz*] | [F_Yz = *f_yz*])

Summary

Value from a string function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
f_xx	“string”	–
f_yy	“string”	–
f_xy	“string”	0.0
f_zz	“string”	–
f_xz	“string”	0.0
f_yz	“string”	0.0

Cvfem_Energy

Constant

Syntax

Bc Flux For Cvfem_Energy ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Data = *data*] | *{flux | value}* = *flux*)

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
data	“string”	–
flux	real	–

Exponential

Syntax

Bc Flux For Cvfem_Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Data = *data*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
data	“string”	–
variable	“string”	–
constant	real	0
multiplier	real	1
exponent	real	–

Exponential_Vapor_Cooling

Syntax

Bc Flux For Cvfem_Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential_Vapor_Cooling ([Data = *data*] | Tboil = *tboil* | L = *l* | Pa = *pa* | Mw = *mw* | Alpha = *alpha* | [Tl = *tl*] | [Th = *th*])

Summary

Exponential vapor cooling energy flux

Description

Flux is given by:

$$F = P_a L\alpha \exp\left(\frac{L}{R}\left(\frac{1}{T_{boil}} - \frac{1}{T}\right)\right)\sqrt{\frac{1}{2\pi RT}}$$

where R is the specific gas constant ($\bar{R}$/MW). For T less than T_L, zero flux is applied. For T between T_L and T_H, a 3rd order smoothing polynomial is used to provide continuity of values and derivatives between 0 and the expression given above. For T greater than T_H, the above expression is used. If not provided, T_L and T_H are set to 10% below and above T_{Boil}, respectively.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
data	“string”	–
tboil	real	–
l	real	–
pa	real	–
mw	real	–
alpha	real	–
tl	real	-1
th	real	-1

Generalized_Nat_Conv

Syntax

Bc Flux For Cvfem_Energy ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Generalized_Nat_Conv ([Data = *data*] | [Multiplier = *multiplier*] | [H = *h*] | [T_Ref = *t_ref*])

Summary

CVFEM energy generalized natural convection flux BC

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
data	"string"	–
multiplier	real	1
h	"string"	–
t_ref	"string"	–

Generalized_Rad

Syntax

Bc Flux For Cvfem_Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Generalized_Rad ([Data = *data*] | [Multiplier = *multiplier*] | [T_Ref = *t_ref*])

Summary

Radiative energy flux, with values for emissivity, form factor and irradiation taken from expressions

Description

Flux is:

$$M\epsilon f\left(\sigma T^4 - I\right)$$

M is MULTIPLIER. σ is the Stefan-Boltzmann constant. ϵ and F are expressions for emissivity and form factor. T is surface temperature. I is the expression for irradiation. If I does not already exist, it will be constructed as σT_{ref}^4, where T_{ref} is either provided by the BC Rad Reference Temperature expression or the inline T_{ref} parameter (constant or string function).

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
data	"string"	–
multiplier	real	1
t_ref	"string"	–

Global

Syntax

Bc Flux For Cvfem_Energy ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Data = *data*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
data	"string"	–
global_name	"string"	–
component	integer	0

Interface

Syntax

Bc Flux For Cvfem_Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Interface [Data = *data*]

Summary

INTERFACE CVFEM energy flux

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
data	“string”	–

Mass_Flux

Syntax

Bc Flux For Cvfem_Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Mass_Flux ([Data = *data*] | Inflow_Phi = *Inflow_Phi* | Total_Mdot = *total_mdot*)

Summary

CVFEM energy mass flux BC

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
data	"string"	–
Inflow_Phi	real	–
total_mdot	real	–

Mass_Flux_From_Temperature

Syntax

Bc Flux For Cvfem_Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Mass_Flux_From_Temperature ([Data = *data*] | [Temperature = *temperature*] | [Pressure = *pressure*] | [Mass_Fraction_0 = *mass_fraction_0*] | [Mass_Fraction_1 = *mass_fraction_1*] | [Mass_Fraction_2 = *mass_fraction_2*] | [Mass_Fraction_3 = *mass_fraction_3*] | [Mass_Fraction_4 = *mass_fraction_4*] | [Mass_Fraction_5 = *mass_fraction_5*] | [Mass_Fraction_6 = *mass_fraction_6*] | [Mass_Fraction_7 = *mass_fraction_7*] | [Mass_Fraction_8 = *mass_fraction_8*] | Total_Mdot = *total_mdot*)

Summary

CVFEM energy mass flux from temperature flux BC

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
data	"string"	–
temperature	real	0
pressure	real	0
mass_fraction_0	real	0
mass_fraction_1	real	0
mass_fraction_2	real	0
mass_fraction_3	real	0
mass_fraction_4	real	0
mass_fraction_5	real	0
mass_fraction_6	real	0
mass_fraction_7	real	0
mass_fraction_8	real	0
total_mdot	real	–

Nat_Conv

Syntax

Bc Flux For Cvfem_Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nat_Conv ([Data = *data*] | H = *h* | T_Ref = *t_ref*)

Summary

CVFEM convective energy flux defined with constant models for heat transfer coefficient and reference temperature

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
data	"string"	–
h	real	–
t_ref	real	–

Nd_Table

Syntax

Bc Flux For Cvfem_Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Data = *data*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
data	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Open_Adv_Flow

Syntax

Bc Flux For Cvfem_Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Open_Adv_Flow ([Data = *data*] | Far_Field_Entrainment_Value = *Far_Field_Entrainment_Value* | [Pressure = *pressure*] | [Total_Pressure = *total_pressure*])

Summary

CVFEM energy open advective flow flux

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
data	"string"	–
Far_Field_Entrainment_Value	real	–
pressure	real	0
total_pressure	real	0

Open_Adv_Flow_From_Temperature

Syntax

Bc Flux For Cvfem_Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Open_Adv_Flow_From_Temperature ([Data = *data*] | [Temperature = *temperature*] | [Pressure = *pressure*] | [Mass_Fraction_0 = *mass_fraction_0*] | [Mass_Fraction_1 = *mass_fraction_1*] | [Mass_Fraction_2 = *mass_fraction_2*] | [Mass_Fraction_3 = *mass_fraction_3*] | [Mass_Fraction_4 = *mass_fraction_4*] | [Mass_Fraction_5 = *mass_fraction_5*] | [Mass_Fraction_6 = *mass_fraction_6*] | [Mass_Fraction_7 = *mass_fraction_7*] | [Mass_Fraction_8 = *mass_fraction_8*] | [Total_Pressure = *total_pressure*])

Summary

CVFEM energy open advective flow from temperature flux

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
data	“string”	–
temperature	real	0
pressure	real	0
mass_fraction_0	real	0
mass_fraction_1	real	0
mass_fraction_2	real	0
mass_fraction_3	real	0
mass_fraction_4	real	0
mass_fraction_5	real	0
mass_fraction_6	real	0
mass_fraction_7	real	0
mass_fraction_8	real	0
total_pressure	real	0

Polynomial

Syntax

Bc Flux For Cvfem_Energy ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Data = *data*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
data	“string”	–
variable	“string”	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Porous_Robin_Coupled

Syntax

Bc Flux For Cvfem_Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Porous_Robin_Coupled ([Data = *data*] | Enthalpy_Field = *enthalpy_field* | Enthalpy_Diffusive_Flux_Field = *enthalpy_diffusive_flux_field* | Averaged_Coeff_Field = *averaged_coeff_field*)

Summary

Boundary condition for loose coupling of enthalpy to a porous region

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
data	"string"	–
enthalpy_field	"string"	–
enthalpy_diffusive_flux_field	"string"	–
averaged_coeff_field	"string"	–

Porous_Robin_Coupled_One_Region

Syntax

Bc Flux For Cvfem_Energy ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Porous_Robin_Coupled_One_Region [Data = *data*]

Summary

Boundary condition for loose coupling of enthalpy to a porous region

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
data	"string"	–

Porous_Robin_Coupled_With_Solid_Phase_Convection

Syntax

Bc Flux For Cvfem_Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Porous_Robin_Coupled_With_Solid_Phase_Convection ([Data = *data*] | [Multiplier = *multiplier*] | Enthalpy_Field = *enthalpy_field* | Enthalpy_Diffusive_Flux_Field = *enthalpy_diffusive_flux_field* | Solid_Phase_Temperature_Field = *solid_phase_temperature_field* | Porosity_Field = *porosity_field* | Averaged_Coeff_Field = *averaged_coeff_field* | Volumetric_Heat_Transfer_Coeff_Field = *volumetric_heat_transfer_coeff_field* | Specific_Surface_Area_Field = *specific_surface_area_field*)

Summary

Boundary condition for loose coupling of enthalpy to a porous region

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
data	“string”	–
multiplier	real	1
enthalpy_field	“string”	–
enthalpy_diffusive_flux_field	“string”	–
solid_phase_temperature_field	“string”	–
porosity_field	“string”	–
averaged_coeff_field	“string”	–
volumetric_heat_transfer_coeff_field	“string”	–
specific_surface_area_field	“string”	–

Porous_Robin_Coupled_With_Solid_Phase_Convection_One_Region

Syntax

Bc Flux For Cvfem_Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Porous_Robin_Coupled_With_Solid_Phase_Convection_One_Region ([Data = *data*] | [Multiplier = *multiplier*])

Summary

Boundary condition for coupling of enthalpy to a porous region in the same region.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
data	“string”	–
multiplier	real	1

Rad

Syntax

Bc Flux For Cvfem_Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Rad ([Data = *data*] | Crad = *crad* | T_Ref = *t_ref*)

Summary

Radiative energy flux defined with constant models for emissivity and reference temperature.

Description

The applied flux is given by:

$$c_{rad}\left(T^4 - T_{ref}^4\right)$$

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
data	"string"	–
crad	real	–
t_ref	real	–

Ray_Trace

Syntax

Bc Flux For Cvfem_Energy ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Ray_Trace ([Data = *data*] | Ray_Source = *ray_source* | [Emission = *emission*])

Summary

Apply a ray tracing energy flux

Description

Adds an energy flux based on the intersection of rays defined in a ray tracing source block for the equation system with name given by the RAY_SOURCE parameter with the surface. The optional EMISSION parameter also imposes a σT^4 blackbody radiative emission term

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
data	"string"	–
ray_source	"string"	–
emission	"string"	FALSE

Scalar_String_Function

Syntax

Bc Flux For Cvfem_Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Data = *data*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
data	“string”	–
f	“string”	–

User_Field

Syntax

Bc Flux For Cvfem_Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Data = *data*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
data	"string"	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc Flux For Cvfem_Energy ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Data = *data*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
data	“string”	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Bc Flux For Cvfem_Energy ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Data = *data*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
data	“string”	–
Name	“string”	–
plugin_parameters	“string”	–

Wall_Function_From_Cht_Temperature

Syntax

Bc Flux For Cvfem_Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Wall_Function_From_Cht_Temperature ([Data = *data*] | Wall_Friction_Factor = *wall_friction_factor*)

Summary

CVFEM energy wall function from conjugate heat transfer temperature flux BC

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
data	“string”	–
wall_friction_factor	real	–

Wall_Function_From_Temperature

Syntax

Bc Flux For Cvfem_Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Wall_Function_From_Temperature ([Data = *data*] | *{t | value}* = *t* | Wall_Friction_Factor = *wall_friction_factor*)

Summary

CVFEM energy wall function from temperature flux BC

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
data	"string"	–
t	real	–
wall_friction_factor	real	–

Cvfem_Level_Set

Open_Adv_Flow

Syntax

Bc Flux For Cvfem_Level_Set ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Open_Adv_Flow ([Toggle = *toggle*] | [Pressure = *pressure*] | [Total_Pressure = *total_pressure*] | Far_Field_Entrainment_Value = *far_field_entrainment_value*)

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
pressure	real	0
total_pressure	real	0
far_field_entrainment_value	real	–

Open_Nc_Adv_Flow

Syntax

Bc Flux For Cvfem_Level_Set ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Open_Nc_Adv_Flow ([Toggle = *toggle*] | Far_Field_Entrainment_Value = *far_field_entrainment_value*)

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
far_field_entrainment_value	real	–

Cvfem_Lumped_Div_Projection

Constant

Syntax

Bc Flux For Cvfem_Lumped_Div_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | [Valuexx = *valuexx*] | [Valuexy = *valuexy*] | [Valuexz = *valuexz*] | [Valueyx = *valueyx*] | [Valueyy = *valueyy*] | [Valueyz = *valueyz*] | [Valuezx = *valuezx*] | [Valuezy = *valuezy*] | [Valuezz = *valuezz*] | [Valuet11 = *valuet11*] | [Valuet12 = *valuet12*] | [Valuet13 = *valuet13*] | [Valuet21 = *valuet21*] | [Valuet22 = *valuet22*] | [Valuet23 = *valuet23*] | [Valuet31 = *valuet31*] | [Valuet32 = *valuet32*] | [Valuet33 = *valuet33*])

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
valuexx	real	0
valuexy	real	0
valuexz	real	0
valueyx	real	0
valueyy	real	0
valueyz	real	0
valuezx	real	0
valuezy	real	0
valuezz	real	0
valuet11	real	0
valuet12	real	0
valuet13	real	0
valuet21	real	0
valuet22	real	0
valuet23	real	0
valuet31	real	0
valuet32	real	0
valuet33	real	0

Simple_Interp_Tensor

Syntax

Bc Flux For Cvfem_Lumped_Div_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Simple_Interp_Tensor ([Toggle = *toggle*] | Source = *source*)

Summary

CVFEM lumped divergence projection simple interpolation tensor flux BC

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
source	"string"	–

Simple_Interp_Vector

Syntax

Bc Flux For Cvfem_Lumped_Div_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Simple_Interp_Vector ([Toggle = *toggle*] | Source = *source*)

Summary

CVFEM lumped divergence projection simple interpolation vector flux BC

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
source	"string"	–

Tensor_String_Function

Syntax

Bc Flux For Cvfem_Lumped_Div_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Tensor_String_Function ([Toggle = *toggle*] | F_Xx = *f_xx* | F_Yy = *f_yy* | [F_Xy = *f_xy*] | [F_Zz = *f_zz*] | [F_Xz = *f_xz*] | [F_Yz = *f_yz*] | [F_Yx = *f_yx*] | [F_Zx = *f_zx*] | [F_Zy = *f_zy*])

Summary

Value from a string function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
f_xx	“string”	–
f_yy	“string”	–
f_xy	“string”	0.0
f_zz	“string”	–
f_xz	“string”	0.0
f_yz	“string”	0.0
f_yx	“string”	0.0
f_zx	“string”	0.0
f_zy	“string”	0.0

Tensor_String_Function_Symmetric

Syntax

Bc Flux For Cvfem_Lumped_Div_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Tensor_String_Function_Symmetric ([Toggle = *toggle*] | F_Xx = *f_xx* | F_Yy = *f_yy* | [F_Xy = *f_xy*] | [F_Zz = *f_zz*] | [F_Xz = *f_xz*] | [F_Yz = *f_yz*])

Summary

Value from a string function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
f_xx	“string”	–
f_yy	“string”	–
f_xy	“string”	0.0
f_zz	“string”	–
f_xz	“string”	0.0
f_yz	“string”	0.0

Cvfem_Lumped_Projection

Constant

Syntax

Bc Flux For Cvfem_Lumped_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | Value = *value*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
value	real	–

Exponential

Syntax

Bc Flux For Cvfem_Lumped_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Flux For Cvfem_Lumped_Projection ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Bc Flux For Cvfem_Lumped_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Open

Syntax

Bc Flux For Cvfem_Lumped_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Open ([Toggle = *toggle*] | [Total_Pressure = *total_pressure*] | [Pressure = *pressure*])

Summary

CVFEM lumped projection open flux BC

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
total_pressure	real	0
pressure	real	0

Polynomial

Syntax

Bc Flux For Cvfem_Lumped_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Flux For Cvfem_Lumped_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f	"string"	–

Simple_Interp

Syntax

Bc Flux For Cvfem_Lumped_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Simple_Interp [Toggle = *toggle*]

Summary

CVFEM lumped projection simple interpolation flux BC

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–

User_Field

Syntax

Bc Flux For Cvfem_Lumped_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
scaling	real	1
global_var	“string”	–
time_function	“string”	–

User_Function

Syntax

Bc Flux For Cvfem_Lumped_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc Flux For Cvfem_Lumped_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

Cvfem_Mass_Fraction

Constant

Syntax

Bc Flux For Cvfem_Mass_Fraction ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | Value = *value*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
value	real	–

Exponential

Syntax

Bc Flux For Cvfem_Mass_Fraction ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Flux For Cvfem_Mass_Fraction ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
global_name	"string"	–
component	integer	0

Mass_Flux

Syntax

Bc Flux For Cvfem_Mass_Fraction ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Mass_Flux ([Toggle = *toggle*] | Total_Mdot = *total_mdot* | [Inflow_Phi = *inflow_phi*])

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
total_mdot	real	–
inflow_phi	real	0

Nd_Table

Syntax

Bc Flux For Cvfem_Mass_Fraction ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Open_Adv_Flow

Syntax

Bc Flux For Cvfem_Mass_Fraction ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Open_Adv_Flow ([Toggle = *toggle*] | [Pressure = *pressure*] | [Total_Pressure = *total_pressure*] | [Far_Field_Entrainment_Value = *far_field_entrainment_value*])

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
pressure	real	0
total_pressure	real	0
far_field_entrainment_value	real	0

Polynomial

Syntax

Bc Flux For Cvfem_Mass_Fraction ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
variable	“string”	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Porous_Robin_Coupled

Syntax

Bc Flux For Cvfem_Mass_Fraction ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Porous_Robin_Coupled ([Toggle = *toggle*] | Mass_Fraction_Field = *mass_fraction_field* | Mass_Fraction_Diffusive_Flux_Field = *mass_fraction_diffusive_flux_field* | Averaged_Coeff_Field = *averaged_coeff_field*)

Summary

Boundary condition for loose coupling of scalars to a free fluid region

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
mass_fraction_field	“string”	–
mass_fraction_diffusive_flux_field	“string”	–
averaged_coeff_field	“string”	–

Porous_Robin_Coupled_One_Region

Syntax

Bc Flux For Cvfem_Mass_Fraction ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Porous_Robin_Coupled_One_Region [Toggle = *toggle*]

Summary

Boundary condition for loose coupling of scalars to a free fluid region

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–

Scalar_String_Function

Syntax

Bc Flux For Cvfem_Mass_Fraction ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f	"string"	–

User_Field

Syntax

Bc Flux For Cvfem_Mass_Fraction ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc Flux For Cvfem_Mass_Fraction ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc Flux For Cvfem_Mass_Fraction ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

Cvfem_Mixture_Fraction

Constant

Syntax

Bc Flux For Cvfem_Mixture_Fraction ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | Value = *value*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
value	real	–

Exponential

Syntax

Bc Flux For Cvfem_Mixture_Fraction ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Free_Open_Flow

Syntax

Bc Flux For Cvfem_Mixture_Fraction ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Free_Open_Flow [Toggle = *toggle*]

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–

Global

Syntax

Bc Flux For Cvfem_Mixture_Fraction ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
global_name	“string”	–
component	integer	0

Mass_Flux

Syntax

Bc Flux For Cvfem_Mixture_Fraction ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Mass_Flux ([Toggle = *toggle*] | Total_Mdot = *total_mdot* | [Inflow_Phi = *inflow_phi*])

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
total_mdot	real	–
inflow_phi	real	0

Nd_Table

Syntax

Bc Flux For Cvfem_Mixture_Fraction ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Open_Adv_Flow

Syntax

Bc Flux For Cvfem_Mixture_Fraction ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Open_Adv_Flow ([Toggle = *toggle*] | [Pressure = *pressure*] | [Total_Pressure = *total_pressure*] | [Far_Field_Entrainment_Value = *far_field_entrainment_value*])

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
pressure	real	0
total_pressure	real	0
far_field_entrainment_value	real	0

Polynomial

Syntax

Bc Flux For Cvfem_Mixture_Fraction ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Flux For Cvfem_Mixture_Fraction ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f	"string"	–

User_Field

Syntax

Bc Flux For Cvfem_Mixture_Fraction ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc Flux For Cvfem_Mixture_Fraction ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc Flux For Cvfem_Mixture_Fraction ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

Cvfem_Momentum

Constant

Syntax

Bc Flux For Cvfem_Momentum ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | [Valuex = *valuex*] | [Valuey = *valuey*] | [Valuez = *valuez*])

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
valuex	real	0
valuey	real	0
valuez	real	0

Constant_Traction

Syntax

Bc Flux For Cvfem_Momentum ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant_Traction ([Toggle = *toggle*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
x	real	0
y	real	0
z	real	0

Exponential_Vapor_Recoil_Pressure

Syntax

Bc Flux For Cvfem_Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential_Vapor_Recoil_Pressure ([Toggle = *toggle*] | Tboil = *tboil* | L = *l* | Pa = *pa* | [Beta = *beta*] | Mw = *mw* | [Tl = *tl*] | [Th = *th*])

Summary

Exponential vapor recoil pressure momentum flux

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
tboil	real	–
l	real	–
pa	real	–
beta	real	1
mw	real	–
tl	real	-1
th	real	-1

Free_Open_Flow

Syntax

Bc Flux For Cvfem_Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Free_Open_Flow ([Toggle = *toggle*] | Pressure = *pressure*)

Summary

Open flow for momentum; apply the full stress

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
pressure	real	–

Interface

Syntax

Bc Flux For Cvfem_Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Interface [Toggle = *toggle*]

Summary

INTERFACE CVFEM momentum flux

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–

Mass_Flux

Syntax

Bc Flux For Cvfem_Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Mass_Flux ([Toggle = *toggle*] | Total_Mdot = *total_mdot* | [Temperature = *temperature*] | [Pressure = *pressure*] | [Mass_Fraction_0 = *mass_fraction_0*] | [Mass_Fraction_1 = *mass_fraction_1*] | [Mass_Fraction_2 = *mass_fraction_2*] | [Mass_Fraction_3 = *mass_fraction_3*] | [Mass_Fraction_4 = *mass_fraction_4*] | [Mass_Fraction_5 = *mass_fraction_5*] | [Mass_Fraction_6 = *mass_fraction_6*] | [Mass_Fraction_7 = *mass_fraction_7*] | [Mass_Fraction_8 = *mass_fraction_8*])

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
total_mdot	real	–
temperature	real	0
pressure	real	0
mass_fraction_0	real	0
mass_fraction_1	real	0
mass_fraction_2	real	0
mass_fraction_3	real	0
mass_fraction_4	real	0
mass_fraction_5	real	0
mass_fraction_6	real	0
mass_fraction_7	real	0
mass_fraction_8	real	0

Open_Adv_Flow

Syntax

Bc Flux For Cvfem_Momentum ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Open_Adv_Flow ([Toggle = *toggle*] | [Pressure = *pressure*] | [Total_Pressure = *total_pressure*] | [Coflow_Entrainment_Value_X = *coflow_entrainment_value_x*] | [Coflow_Entrainment_Value_Y = *coflow_entrainment_value_y*] | [Coflow_Entrainment_Value_Z = *coflow_entrainment_value_z*])

Summary

CVFEM momentum open advective flow flux BC

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is

active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
pressure	real	0
total_pressure	real	0
coflow_entrainment_value_x	real	0
coflow_entrainment_value_y	real	0
coflow_entrainment_value_z	real	0

Open_Flow

Syntax

Bc Flux For Cvfem_Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Open_Flow ([Toggle = *toggle*] | [Pressure = *pressure*] | [Total_Pressure = *total_pressure*])

Summary

Open flow for momentum; enforce normal gradient to be zero

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
pressure	real	0
total_pressure	real	0

Slip

Syntax

Bc Flux For Cvfem_Momentum ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Slip ([Toggle = *toggle*] | [Vs_X = *vs_x*] | [Vs_Y = *vs_y*] | [Vs_Z = *vs_z*] | Beta = *beta*)

Summary

Slip momentum flux

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
vs_x	real	0
vs_y	real	0
vs_z	real	0
beta	real	–

Slip_Length

Syntax

Bc Flux For Cvfem_Momentum ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Slip_Length ([Toggle = *toggle*] | [Vs_X = *vs_x*] | [Vs_Y = *vs_y*] | [Vs_Z = *vs_z*] | Beta = *beta*)

Summary

Slip length momentum flux

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
vs_x	real	0
vs_y	real	0
vs_z	real	0
beta	real	–

Symmetry_Flow

Syntax

Bc Flux For Cvfem_Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Symmetry_Flow [Toggle = *toggle*]

Summary

CVFEM momentum symmetry flow flux BC

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–

Vector_String_Function

Syntax

Bc Flux For Cvfem_Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Vector_String_Function ([Toggle = *toggle*] | F_X = *f_x* | F_Y = *f_y* | [F_Z = *f_z*])

Summary

Value from a string function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f_x	"string"	–
f_y	"string"	–
f_z	"string"	–

Wall_Function

Syntax

Bc Flux For Cvfem_Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Wall_Function ([Toggle = *toggle*] | [Wall_Velocity_X = *wall_velocity_x*] | [Wall_Velocity_Y = *wall_velocity_y*] | [Wall_Velocity_Z = *wall_velocity_z*] | Wall_Friction_Factor = *wall_friction_factor*)

Summary

CVFEM momentum wall function flux BC

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
wall_velocity_x	real	0
wall_velocity_y	real	0
wall_velocity_z	real	0
wall_friction_factor	real	–

Cvfem_Momentum_X

Constant

Syntax

Bc Flux For Cvfem_Momentum_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent*

Name ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | Value = *value*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
value	real	–

Exponential

Syntax

Bc Flux For Cvfem_Momentum_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Flux For Cvfem_Momentum_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Bc Flux For Cvfem_Momentum_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc Flux For Cvfem_Momentum_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Flux For Cvfem_Momentum_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f	"string"	–

User_Field

Syntax

Bc Flux For Cvfem_Momentum_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc Flux For Cvfem_Momentum_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc Flux For Cvfem_Momentum_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

Cvfem_Momentum_Y

Constant

Syntax

Bc Flux For Cvfem_Momentum_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | Value = *value*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
value	real	–

Exponential

Syntax

Bc Flux For Cvfem_Momentum_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Flux For Cvfem_Momentum_Y ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Bc Flux For Cvfem_Momentum_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc Flux For Cvfem_Momentum_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Flux For Cvfem_Momentum_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f	"string"	–

User_Field

Syntax

Bc Flux For Cvfem_Momentum_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc Flux For Cvfem_Momentum_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc Flux For Cvfem_Momentum_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

Cvfem_Momentum_Z

Constant

Syntax

Bc Flux For Cvfem_Momentum_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | Value = *value*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
value	real	–

Exponential

Syntax

Bc Flux For Cvfem_Momentum_Z ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Flux For Cvfem_Momentum_Z ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Bc Flux For Cvfem_Momentum_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc Flux For Cvfem_Momentum_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Flux For Cvfem_Momentum_Z ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f	"string"	–

User_Field

Syntax

Bc Flux For Cvfem_Momentum_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc Flux For Cvfem_Momentum_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc Flux For Cvfem_Momentum_Z ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

Cvfem_Pressure_Edge_Gradient_Projection

Constant

Syntax

Bc Flux For Cvfem_Pressure_Edge_Gradient_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | Value = *value*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
value	real	–

Exponential

Syntax

Bc Flux For Cvfem_Pressure_Edge_Gradient_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Flux For Cvfem_Pressure_Edge_Gradient_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Bc Flux For Cvfem_Pressure_Edge_Gradient_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc Flux For Cvfem_Pressure_Edge_Gradient_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Flux For Cvfem_Pressure_Edge_Gradient_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f	"string"	–

Simple_Interp

Syntax

Bc Flux For Cvfem_Pressure_Edge_Gradient_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Simple_Interp [Toggle = *toggle*]

Summary

CVFEM edge-based projection simple interpolation flux BC

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–

User_Field

Syntax

Bc Flux For Cvfem_Pressure_Edge_Gradient_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
scaling	real	1
global_var	“string”	–
time_function	“string”	–

User_Function

Syntax

Bc Flux For Cvfem_Pressure_Edge_Gradient_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc Flux For Cvfem_Pressure_Edge_Gradient_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

Cvfem_Projection

Constant

Syntax

Bc Flux For Cvfem_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | Value = *value*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
value	real	–

Exponential

Syntax

Bc Flux For Cvfem_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Flux For Cvfem_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Bc Flux For Cvfem_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Open

Syntax

Bc Flux For Cvfem_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Open ([Toggle = *toggle*] | [Total_Pressure = *total_pressure*] | [Pressure = *pressure*])

Summary

CVFEM projection open flux BC

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
total_pressure	real	0
pressure	real	0

Polynomial

Syntax

Bc Flux For Cvfem_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i (X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Flux For Cvfem_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f	"string"	–

Simple_Interp

Syntax

Bc Flux For Cvfem_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Simple_Interp [Toggle = *toggle*]

Summary

CVFEM projection simple interpolation flux BC

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–

User_Field

Syntax

Bc Flux For Cvfem_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
scaling	real	1
global_var	“string”	–
time_function	“string”	–

User_Function

Syntax

Bc Flux For Cvfem_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc Flux For Cvfem_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

Cvfem_Sdr_Edge_Gradient_Projection

Constant

Syntax

Bc Flux For Cvfem_Sdr_Edge_Gradient_Projection ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | Value = *value*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
value	real	–

Exponential

Syntax

Bc Flux For Cvfem_Sdr_Edge_Gradient_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Flux For Cvfem_Sdr_Edge_Gradient_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Bc Flux For Cvfem_Sdr_Edge_Gradient_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc Flux For Cvfem_Sdr_Edge_Gradient_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Flux For Cvfem_Sdr_Edge_Gradient_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f	"string"	–

Simple_Interp

Syntax

Bc Flux For Cvfem_Sdr_Edge_Gradient_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Simple_Interp [Toggle = *toggle*]

Summary

CVFEM edge-based projection simple interpolation flux BC

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–

User_Field

Syntax

Bc Flux For Cvfem_Sdr_Edge_Gradient_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
scaling	real	1
global_var	“string”	–
time_function	“string”	–

User_Function

Syntax

Bc Flux For Cvfem_Sdr_Edge_Gradient_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc Flux For Cvfem_Sdr_Edge_Gradient_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

Cvfem_Solvent_Continuity

Constant

Syntax

Bc Flux For Cvfem_Solvent_Continuity ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | Value = *value*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
value	real	–

Exponential

Syntax

Bc Flux For Cvfem_Solvent_Continuity ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Flux For Cvfem_Solvent_Continuity ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
global_name	"string"	–
component	integer	0

Inflow

Syntax

Bc Flux For Cvfem_Solvent_Continuity ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Inflow [Toggle = *toggle*]

Summary

Inflow mass flux for continuity equation; nodal interpolation

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–

Nd_Table

Syntax

Bc Flux For Cvfem_Solvent_Continuity ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Open_Flow

Syntax

Bc Flux For Cvfem_Solvent_Continuity ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Open_Flow ([Toggle = *toggle*] | [Pressure = *pressure*] | [Total_Pressure = *total_pressure*])

Summary

Open bc for continuity

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
pressure	real	0
total_pressure	real	0

Polynomial

Syntax

Bc Flux For Cvfem_Solvent_Continuity ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
variable	“string”	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Flux For Cvfem_Solvent_Continuity ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f	"string"	–

Simple_Inflow

Syntax

Bc Flux For Cvfem_Solvent_Continuity ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Simple_Inflow ([Toggle = *toggle*] | *{density | value}* = *Density* | [Velocity_X = *Velocity_X*] | [Velocity_Y = *Velocity_Y*] | [Velocity_Z = *Velocity_Z*])

Summary

Inflow mass flux for continuity equation; specified density and velocity

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Density	real	–
Velocity_X	real	0
Velocity_Y	real	0
Velocity_Z	real	0

User_Field

Syntax

Bc Flux For Cvfem_Solvent_Continuity ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc Flux For Cvfem_Solvent_Continuity ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc Flux For Cvfem_Solvent_Continuity ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

Cvfem_Solvent_Momentum

Constant

Syntax

Bc Flux For Cvfem_Solvent_Momentum ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | [Valuex = *valuex*] | [Valuey = *valuey*] | [Valuez = *valuez*])

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
valuex	real	0
valuey	real	0
valuez	real	0

Open_Adv_Flow

Syntax

Bc Flux For Cvfem_Solvent_Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Open_Adv_Flow ([Toggle = *toggle*] | [Pressure = *pressure*] | [Total_Pressure = *total_pressure*] | [Coflow_Entrainment_Value_X = *coflow_entrainment_value_x*] | [Coflow_Entrainment_Value_Y = *coflow_entrainment_value_y*] | [Coflow_Entrainment_Value_Z = *coflow_entrainment_value_z*])

Summary

CVFEM solvent momentum open advective flow flux BC

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
pressure	real	0
total_pressure	real	0
coflow_entrainment_value_x	real	0
coflow_entrainment_value_y	real	0
coflow_entrainment_value_z	real	0

Open_Flow

Syntax

Bc Flux For Cvfem_Solvent_Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Open_Flow ([Toggle = *toggle*] | [Pressure = *pressure*] | [Total_Pressure = *total_pressure*])

Summary

CVFEM solvent momentum open flow flux BC

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
pressure	real	0
total_pressure	real	0

Symmetry_Flow

Syntax

Bc Flux For Cvfem_Solvent_Momentum ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Symmetry_Flow [Toggle = *toggle*]

Summary

CVFEM solvent momentum symmetry flow flux BC

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–

Vector_String_Function

Syntax

Bc Flux For Cvfem_Solvent_Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Vector_String_Function ([Toggle = *toggle*] | F_X = *f_x* | F_Y = *f_y* | [F_Z = *f_z*])

Summary

Value from a string function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
f_x	“string”	–
f_y	“string”	–
f_z	“string”	–

Cvfem_Solvent_Momentum_X

Constant

Syntax

Bc Flux For Cvfem_Solvent_Momentum_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | Value = *value*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
value	real	–

Exponential

Syntax

Bc Flux For Cvfem_Solvent_Momentum_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Flux For Cvfem_Solvent_Momentum_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Bc Flux For Cvfem_Solvent_Momentum_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc Flux For Cvfem_Solvent_Momentum_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Flux For Cvfem_Solvent_Momentum_X ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f	"string"	–

User_Field

Syntax

Bc Flux For Cvfem_Solvent_Momentum_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc Flux For Cvfem_Solvent_Momentum_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc Flux For Cvfem_Solvent_Momentum_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

Cvfem_Solvent_Momentum_Y

Constant

Syntax

Bc Flux For Cvfem_Solvent_Momentum_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | Value = *value*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
value	real	–

Exponential

Syntax

Bc Flux For Cvfem_Solvent_Momentum_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Flux For Cvfem_Solvent_Momentum_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Bc Flux For Cvfem_Solvent_Momentum_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc Flux For Cvfem_Solvent_Momentum_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Flux For Cvfem_Solvent_Momentum_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f	"string"	–

User_Field

Syntax

Bc Flux For Cvfem_Solvent_Momentum_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc Flux For Cvfem_Solvent_Momentum_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc Flux For Cvfem_Solvent_Momentum_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

Cvfem_Solvent_Momentum_Z

Constant

Syntax

Bc Flux For Cvfem_Solvent_Momentum_Z ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | Value = *value*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
value	real	–

Exponential

Syntax

Bc Flux For Cvfem_Solvent_Momentum_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Flux For Cvfem_Solvent_Momentum_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Bc Flux For Cvfem_Solvent_Momentum_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc Flux For Cvfem_Solvent_Momentum_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Flux For Cvfem_Solvent_Momentum_Z ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f	"string"	–

User_Field

Syntax

Bc Flux For Cvfem_Solvent_Momentum_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc Flux For Cvfem_Solvent_Momentum_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc Flux For Cvfem_Solvent_Momentum_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

Cvfem_Specific_Dissipation_Rate

Constant

Syntax

Bc Flux For Cvfem_Specific_Dissipation_Rate ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | Value = *value*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
value	real	–

Exponential

Syntax

Bc Flux For Cvfem_Specific_Dissipation_Rate ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Flux For Cvfem_Specific_Dissipation_Rate ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
global_name	"string"	–
component	integer	0

Mass_Flux

Syntax

Bc Flux For Cvfem_Specific_Dissipation_Rate ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Mass_Flux ([Toggle = *toggle*] | Total_Mdot = *total_mdot* | [Inflow_Phi = *inflow_phi*])

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
total_mdot	real	–
inflow_phi	real	0

Nd_Table

Syntax

Bc Flux For Cvfem_Specific_Dissipation_Rate ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Open_Adv_Flow

Syntax

Bc Flux For Cvfem_Specific_Dissipation_Rate ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Open_Adv_Flow ([Toggle = *toggle*] | [Pressure = *pressure*] | [Total_Pressure = *total_pressure*] | [Far_Field_Entrainment_Value = *far_field_entrainment_value*])

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
pressure	real	0
total_pressure	real	0
far_field_entrainment_value	real	0

Polynomial

Syntax

Bc Flux For Cvfem_Specific_Dissipation_Rate ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
variable	“string”	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Flux For Cvfem_Specific_Dissipation_Rate ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f	"string"	–

User_Field

Syntax

Bc Flux For Cvfem_Specific_Dissipation_Rate ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc Flux For Cvfem_Specific_Dissipation_Rate ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc Flux For Cvfem_Specific_Dissipation_Rate ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

Cvfem_Temperature_Edge_Gradient_Projection

Constant

Syntax

Bc Flux For Cvfem_Temperature_Edge_Gradient_Projection ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | Value = *value*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
value	real	–

Exponential

Syntax

Bc Flux For Cvfem_Temperature_Edge_Gradient_Projection ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Flux For Cvfem_Temperature_Edge_Gradient_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Bc Flux For Cvfem_Temperature_Edge_Gradient_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc Flux For Cvfem_Temperature_Edge_Gradient_Projection ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Flux For Cvfem_Temperature_Edge_Gradient_Projection ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f	"string"	–

Simple_Interp

Syntax

Bc Flux For Cvfem_Temperature_Edge_Gradient_Projection ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Simple_Interp [Toggle = *toggle*]

Summary

CVFEM edge-based projection simple interpolation flux BC

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–

User_Field

Syntax

Bc Flux For Cvfem_Temperature_Edge_Gradient_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
scaling	real	1
global_var	“string”	–
time_function	“string”	–

User_Function

Syntax

Bc Flux For Cvfem_Temperature_Edge_Gradient_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc Flux For Cvfem_Temperature_Edge_Gradient_Projection ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

Cvfem_Tke_Edge_Gradient_Projection

Constant

Syntax

Bc Flux For Cvfem_Tke_Edge_Gradient_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | Value = *value*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
value	real	–

Exponential

Syntax

Bc Flux For Cvfem_Tke_Edge_Gradient_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Flux For Cvfem_Tke_Edge_Gradient_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Bc Flux For Cvfem_Tke_Edge_Gradient_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc Flux For Cvfem_Tke_Edge_Gradient_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Flux For Cvfem_Tke_Edge_Gradient_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f	"string"	–

Simple_Interp

Syntax

Bc Flux For Cvfem_Tke_Edge_Gradient_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Simple_Interp [Toggle = *toggle*]

Summary

CVFEM edge-based projection simple interpolation flux BC

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–

User_Field

Syntax

Bc Flux For Cvfem_Tke_Edge_Gradient_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
scaling	real	1
global_var	“string”	–
time_function	“string”	–

User_Function

Syntax

Bc Flux For Cvfem_Tke_Edge_Gradient_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc Flux For Cvfem_Tke_Edge_Gradient_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

Cvfem_Turbulence_Dissipation_Rate

Constant

Syntax

Bc Flux For Cvfem_Turbulence_Dissipation_Rate ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | Value = *value*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
value	real	–

Exponential

Syntax

Bc Flux For Cvfem_Turbulence_Dissipation_Rate ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Free_Open_Flow

Syntax

Bc Flux For Cvfem_Turbulence_Dissipation_Rate ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Free_Open_Flow [Toggle = *toggle*]

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–

Global

Syntax

Bc Flux For Cvfem_Turbulence_Dissipation_Rate ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
global_name	“string”	–
component	integer	0

Mass_Flux

Syntax

Bc Flux For Cvfem_Turbulence_Dissipation_Rate ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Mass_Flux ([Toggle = *toggle*] | Total_Mdot = *total_mdot* | [Inflow_Phi = *inflow_phi*])

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
total_mdot	real	–
inflow_phi	real	0

Nd_Table

Syntax

Bc Flux For Cvfem_Turbulence_Dissipation_Rate ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Open_Adv_Flow

Syntax

Bc Flux For Cvfem_Turbulence_Dissipation_Rate ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Open_Adv_Flow ([Toggle = *toggle*] | [Pressure = *pressure*] | [Total_Pressure = *total_pressure*] | [Far_Field_Entrainment_Value = *far_field_entrainment_value*])

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
pressure	real	0
total_pressure	real	0
far_field_entrainment_value	real	0

Polynomial

Syntax

Bc Flux For Cvfem_Turbulence_Dissipation_Rate ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Flux For Cvfem_Turbulence_Dissipation_Rate ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f	"string"	–

User_Field

Syntax

Bc Flux For Cvfem_Turbulence_Dissipation_Rate ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc Flux For Cvfem_Turbulence_Dissipation_Rate ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc Flux For Cvfem_Turbulence_Dissipation_Rate ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

Cvfem_Turbulent_Kinetic_Energy

Constant

Syntax

Bc Flux For Cvfem_Turbulent_Kinetic_Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | Value = *value*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
value	real	–

Exponential

Syntax

Bc Flux For Cvfem_Turbulent_Kinetic_Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Free_Open_Flow

Syntax

Bc Flux For Cvfem_Turbulent_Kinetic_Energy ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Free_Open_Flow [Toggle = *toggle*]

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–

Global

Syntax

Bc Flux For Cvfem_Turbulent_Kinetic_Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
global_name	“string”	–
component	integer	0

Mass_Flux

Syntax

Bc Flux For Cvfem_Turbulent_Kinetic_Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Mass_Flux ([Toggle = *toggle*] | Total_Mdot = *total_mdot* | [Inflow_Phi = *inflow_phi*])

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
total_mdot	real	–
inflow_phi	real	0

Nd_Table

Syntax

Bc Flux For Cvfem_Turbulent_Kinetic_Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Open_Adv_Flow

Syntax

Bc Flux For Cvfem_Turbulent_Kinetic_Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Open_Adv_Flow ([Toggle = *toggle*] | [Pressure = *pressure*] | [Total_Pressure = *total_pressure*] | [Far_Field_Entrainment_Value = *far_field_entrainment_value*])

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
pressure	real	0
total_pressure	real	0
far_field_entrainment_value	real	0

Polynomial

Syntax

Bc Flux For Cvfem_Turbulent_Kinetic_Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Flux For Cvfem_Turbulent_Kinetic_Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f	"string"	–

User_Field

Syntax

Bc Flux For Cvfem_Turbulent_Kinetic_Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc Flux For Cvfem_Turbulent_Kinetic_Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc Flux For Cvfem_Turbulent_Kinetic_Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

Cvfem_Velocity_Edge_Gradient_Projection

Constant

Syntax

Bc Flux For Cvfem_Velocity_Edge_Gradient_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | Value = *value*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
value	real	–

Exponential

Syntax

Bc Flux For Cvfem_Velocity_Edge_Gradient_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Flux For Cvfem_Velocity_Edge_Gradient_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Bc Flux For Cvfem_Velocity_Edge_Gradient_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc Flux For Cvfem_Velocity_Edge_Gradient_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Flux For Cvfem_Velocity_Edge_Gradient_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f	"string"	–

User_Field

Syntax

Bc Flux For Cvfem_Velocity_Edge_Gradient_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc Flux For Cvfem_Velocity_Edge_Gradient_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc Flux For Cvfem_Velocity_Edge_Gradient_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

Cvfem_Velocity_Pressure_Projection

Constant

Syntax

Bc Flux For Cvfem_Velocity_Pressure_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | [Valuex = *valuex*] | [Valuey = *valuey*] | [Valuez = *valuez*])

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
valuex	real	0
valuey	real	0
valuez	real	0

Simple_Dp_Interp

Syntax

Bc Flux For Cvfem_Velocity_Pressure_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Simple_Dp_Interp [Toggle = *toggle*]

Summary

Simple DP interpolation model for CVFEM velocity pressure projection flux

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–

Vector_String_Function

Syntax

Bc Flux For Cvfem_Velocity_Pressure_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Vector_String_Function ([Toggle = *toggle*] | F_X = *f_x* | F_Y = *f_y* | [F_Z = *f_z*])

Summary

Value from a string function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
f_x	“string”	–
f_y	“string”	–
f_z	“string”	–

Cvfem_Velocity_Pressure_Projection_X

Constant

Syntax

Bc Flux For Cvfem_Velocity_Pressure_Projection_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | Value = *value*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
value	real	–

Exponential

Syntax

Bc Flux For Cvfem_Velocity_Pressure_Projection_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Flux For Cvfem_Velocity_Pressure_Projection_X ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Bc Flux For Cvfem_Velocity_Pressure_Projection_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc Flux For Cvfem_Velocity_Pressure_Projection_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Flux For Cvfem_Velocity_Pressure_Projection_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f	"string"	–

User_Field

Syntax

Bc Flux For Cvfem_Velocity_Pressure_Projection_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc Flux For Cvfem_Velocity_Pressure_Projection_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc Flux For Cvfem_Velocity_Pressure_Projection_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
Name	“string”	–
plugin_parameters	“string”	–

Cvfem_Velocity_Pressure_Projection_Y

Constant

Syntax

Bc Flux For Cvfem_Velocity_Pressure_Projection_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | Value = *value*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
value	real	–

Exponential

Syntax

Bc Flux For Cvfem_Velocity_Pressure_Projection_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Flux For Cvfem_Velocity_Pressure_Projection_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Bc Flux For Cvfem_Velocity_Pressure_Projection_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc Flux For Cvfem_Velocity_Pressure_Projection_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Flux For Cvfem_Velocity_Pressure_Projection_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f	"string"	–

User_Field

Syntax

Bc Flux For Cvfem_Velocity_Pressure_Projection_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc Flux For Cvfem_Velocity_Pressure_Projection_Y ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc Flux For Cvfem_Velocity_Pressure_Projection_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
Name	“string”	–
plugin_parameters	“string”	–

Cvfem_Velocity_Pressure_Projection_Z

Constant

Syntax

Bc Flux For Cvfem_Velocity_Pressure_Projection_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | Value = *value*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
value	real	–

Exponential

Syntax

Bc Flux For Cvfem_Velocity_Pressure_Projection_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Flux For Cvfem_Velocity_Pressure_Projection_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Bc Flux For Cvfem_Velocity_Pressure_Projection_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc Flux For Cvfem_Velocity_Pressure_Projection_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Flux For Cvfem_Velocity_Pressure_Projection_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f	"string"	–

User_Field

Syntax

Bc Flux For Cvfem_Velocity_Pressure_Projection_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc Flux For Cvfem_Velocity_Pressure_Projection_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc Flux For Cvfem_Velocity_Pressure_Projection_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

Darcy_Momentum

Constant

Syntax

Bc Flux For Darcy_Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | [Valuex = *valuex*] | [Valuey = *valuey*] | [Valuez = *valuez*])

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
valuex	real	0
valuey	real	0
valuez	real	0

Open_Flow

Syntax

Bc Flux For Darcy_Momentum ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Open_Flow ([Toggle = *toggle*] | [Pressure = *pressure*] | [Total_Pressure = *total_pressure*])

Summary

Open flow momentum flux

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
pressure	real	0
total_pressure	real	0

Pressure

Syntax

Bc Flux For Darcy_Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Pressure ([Toggle = *toggle*] | [User_Field_Name = *user_field_name*] | [Use_Bulk_Node = *use_bulk_node*] | [P = *p*] | [C_T = *c_t*] | [C_X = *c_x*] | [C_Y = *c_y*] | [C_Z = *c_z*] | [C_Cos_W = *c_cos_w*] | [C_Sin_W = *c_sin_w*] | [C1_Cos = *c1_cos*] | [C1_Sin = *c1_sin*])

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
user_field_name	"string"	–
use_bulk_node	"string"	–
p	real	0
c_t	real	0
c_x	real	0
c_y	real	0
c_z	real	0
c_cos_w	real	0
c_sin_w	real	0
c1_cos	real	0
c1_sin	real	0

Rt_Pressure

Syntax

Bc Flux For Darcy_Momentum ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Rt_Pressure ([Toggle = *toggle*] | P = *p* | [C_T = *c_t*] | [C_X = *c_x*] | [C_Y = *c_y*] | [C_Z = *c_z*])

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
p	real	–
c_t	real	0
c_x	real	0
c_y	real	0
c_z	real	0

Simple_Interp

Syntax

Bc Flux For Darcy_Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Simple_Interp [Toggle = *toggle*]

Summary

Darcy momentum simple interpolation flux BC

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–

Slip

Syntax

Bc Flux For Darcy_Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Slip ([Toggle = *toggle*] | [Vs_X = *vs_x*] | [Vs_Y = *vs_y*] | [Vs_Z = *vs_z*] | K = *k*)

Summary

Darcy slip momentum flux

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
vs_x	real	0
vs_y	real	0
vs_z	real	0
k	real	–

Vector_String_Function

Syntax

Bc Flux For Darcy_Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Vector_String_Function ([Toggle = *toggle*] | F_X = *f_x* | F_Y = *f_y* | [F_Z = *f_z*])

Summary

Value from a string function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f_x	"string"	–
f_y	"string"	–
f_z	"string"	–

Darcy_Momentum_X

Constant

Syntax

Bc Flux For Darcy_Momentum_X ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | Value = *value*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
value	real	–

Exponential

Syntax

Bc Flux For Darcy_Momentum_X ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Flux For Darcy_Momentum_X ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Bc Flux For Darcy_Momentum_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc Flux For Darcy_Momentum_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Flux For Darcy_Momentum_X ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f	"string"	–

User_Field

Syntax

Bc Flux For Darcy_Momentum_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
scaling	real	1
global_var	“string”	–
time_function	“string”	–

User_Function

Syntax

Bc Flux For Darcy_Momentum_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc Flux For Darcy_Momentum_X ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

Darcy_Momentum_Y

Constant

Syntax

Bc Flux For Darcy_Momentum_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | Value = *value*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
value	real	–

Exponential

Syntax

Bc Flux For Darcy_Momentum_Y ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Flux For Darcy_Momentum_Y ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Bc Flux For Darcy_Momentum_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc Flux For Darcy_Momentum_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Flux For Darcy_Momentum_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f	"string"	–

User_Field

Syntax

Bc Flux For Darcy_Momentum_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc Flux For Darcy_Momentum_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc Flux For Darcy_Momentum_Y ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

Darcy_Momentum_Z

Constant

Syntax

Bc Flux For Darcy_Momentum_Z ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | Value = *value*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
value	real	–

Exponential

Syntax

Bc Flux For Darcy_Momentum_Z ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Flux For Darcy_Momentum_Z ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Bc Flux For Darcy_Momentum_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc Flux For Darcy_Momentum_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Flux For Darcy_Momentum_Z ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f	"string"	–

User_Field

Syntax

Bc Flux For Darcy_Momentum_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc Flux For Darcy_Momentum_Z ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc Flux For Darcy_Momentum_Z ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

Div_Projection

Constant

Syntax

Bc Flux For Div_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | [Valuex = *valuex*] | [Valuey = *valuey*] | [Valuez = *valuez*])

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
valuex	real	0
valuey	real	0
valuez	real	0

Vector_String_Function

Syntax

Bc Flux For Div_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Vector_String_Function ([Toggle = *toggle*] | F_X = *f_x* | F_Y = *f_y* | [F_Z = *f_z*])

Summary

Value from a string function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f_x	"string"	–
f_y	"string"	–
f_z	"string"	–

Div_Projection_X

Constant

Syntax

Bc Flux For Div_Projection_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | Value = *value*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
value	real	–

Exponential

Syntax

Bc Flux For Div_Projection_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Flux For Div_Projection_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Bc Flux For Div_Projection_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc Flux For Div_Projection_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Flux For Div_Projection_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f	"string"	–

User_Field

Syntax

Bc Flux For Div_Projection_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc Flux For Div_Projection_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc Flux For Div_Projection_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

Div_Projection_Y

Constant

Syntax

Bc Flux For Div_Projection_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | Value = *value*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
value	real	–

Exponential

Syntax

Bc Flux For Div_Projection_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Flux For Div_Projection_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Bc Flux For Div_Projection_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc Flux For Div_Projection_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Flux For Div_Projection_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f	"string"	–

User_Field

Syntax

Bc Flux For Div_Projection_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc Flux For Div_Projection_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc Flux For Div_Projection_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

Div_Projection_Z

Constant

Syntax

Bc Flux For Div_Projection_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | Value = *value*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
value	real	–

Exponential

Syntax

Bc Flux For Div_Projection_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Flux For Div_Projection_Z ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Bc Flux For Div_Projection_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc Flux For Div_Projection_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Flux For Div_Projection_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f	"string"	–

User_Field

Syntax

Bc Flux For Div_Projection_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc Flux For Div_Projection_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc Flux For Div_Projection_Z ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

Energy

Calore_User_Sub

Syntax

Bc Flux For Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Calore_User_Sub ([Power_Output = *power_output*] | [Flux_Output = *flux_output*] | [Toggle = *toggle*] | Name = *name* | Type = *type* | [Multiplier = *multiplier*] | [Material_Data_Block = *material_data_block*] | [Data = *data*] | [Scaling_Field = *scaling_field*])

Summary

Values from a Calore user subroutine

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output

If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Flux_Output

If specified, the average flux of the given BC/Src is calculated and stored into the global variable name specified as the *flux_output* argument.

Toggle

If given, specifies which *toggle block* controls whether this BC is

active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
power_output	"string"	–
flux_output	"string"	–
toggle	"string"	–
name	"string"	–
type	"string"	–
multiplier	real	1
material_data_block	"string"	–
data	"string"	–
scaling_field	"string"	–

Cht_Flux

Syntax

Bc Flux For Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Cht_Flux ([Power_Output = *power_output*] | [Flux_Output = *flux_output*] | [Toggle = *toggle*] | [Too_Field = *Too_Field*] | [H_Field = *H_Field*])

Summary

Applies H(T-Too) flux for conjugate heat transfer boundary. Optional parameters are "Too_Field" and "H_Field", which default to "nodal_cht_Too" and "nodal_cht_H" respectively.

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output

If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Flux_Output

If specified, the average flux of the given BC/Src is calculated and stored into the global variable name specified as the *flux_output* argument.

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
power_output	"string"	–
flux_output	"string"	–
toggle	"string"	–
Too_Field	"string"	nodal_cht_Too
H_Field	"string"	nodal_cht_H

Cht_Robin

Syntax

Bc Flux For Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Cht_Robin ([Power_Output = *power_output*] | [Flux_Output = *flux_output*] | [Toggle = *toggle*])

Summary

Applies a Robin-style BC of the form: J dot n + h * (T-Tref).

Description

J, h, and Tref should be specified using the material properties BC_HEAT_FLUX, HEAT_TRANSFER_COEFFICIENT, and BC_REFERENCE_TEMPERATURE in a material that is applied to the surface.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output

If specified, the total (integrated) power of the given BC/Src is

calculated and stored into the global variable name specified as the *power_output* argument.

Flux_Output

If specified, the average flux of the given BC/Src is calculated and stored into the global variable name specified as the *flux_output* argument.

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
power_output	"string"	–
flux_output	"string"	–
toggle	"string"	–

Constant

Syntax

Bc Flux For Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Power_Output = *power_output*] | [Flux_Output = *flux_output*] | [Toggle = *toggle*] | *{flux | value}* = *flux*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output

If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Flux_Output

If specified, the average flux of the given BC/Src is calculated and stored into the global variable name specified as the *flux_output* argument.

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
power_output	"string"	–
flux_output	"string"	–
toggle	"string"	–
flux	real	–

Distribution_Factor

Syntax

Bc Flux For Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Distribution_Factor ([Power_Output = *power_output*] | [Flux_Output = *flux_output*] | [Toggle = *toggle*] | [Attribute_Name = *attribute_name*] | [Multiplier = *multiplier*])

Summary

Applies a flux equal to a distribution factor defined in the input mesh with name ATTRIBUTE_NAME. F = MULTIPLIER x ATTRIBUTE_NAME[node]

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output

If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Flux_Output

If specified, the average flux of the given BC/Src is calculated and stored into the global variable name specified as the *flux_output* argument.

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
power_output	"string"	–
flux_output	"string"	–
toggle	"string"	–
attribute_name	"string"	DISTRIBUTION_FACTOR
multiplier	real	1

Enclosure_Radiation

Syntax

Bc Flux For Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Enclosure_Radiation ([Power_Output = *power_output*] | [Flux_Output = *flux_output*] | [Toggle = *toggle*] | [Multiplier = *multiplier*] | Enclosure = *enclosure*)

Summary

This term imposes an energy flux due to enclosure radiation as defined in *Enclosure Radiation*. The irradiation is provided by the Chapparal radiosity solution

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output

If specified, the total (integrated) power of the given BC/Src is

calculated and stored into the global variable name specified as the *power_output* argument.

Flux_Output

If specified, the average flux of the given BC/Src is calculated and stored into the global variable name specified as the *flux_output* argument.

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
power_output	"string"	–
flux_output	"string"	–
toggle	"string"	–
multiplier	real	1
enclosure	"string"	–

Exponential

Syntax

Bc Flux For Energy ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Power_Output = *power_output*] | [Flux_Output = *flux_output*] | [Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

$$F = C + Me^{kX}$$

where X is the variable, C is the constant, M is the multiplier, and k is exponent. C and M are optional and default to 0 and 1 respectively.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output
If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Flux_Output
If specified, the average flux of the given BC/Src is calculated and stored into the global variable name specified as the *flux_output* argument.

Toggle
If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
power_output	"string"	–
flux_output	"string"	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Exponential_Vapor_Cooling

Syntax

Bc Flux For Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential_Vapor_Cooling ([Power_Output = *power_output*] | [Flux_Output = *flux_output*] | [Toggle = *toggle*] | Tboil = *tboil* | L = *l* | Pa = *pa* | Mw = *mw* | Alpha = *alpha* | [Tl = *tl*] | [Th = *th*])

Summary

Exponential vapor cooling energy flux

Description

Flux is given by:

$$F = P_a L\alpha \exp\left(\frac{L}{R}\left(\frac{1}{T_{boil}} - \frac{1}{T}\right)\right)\sqrt{\frac{1}{2\pi RT}}$$

where R is the specific gas constant ($\bar{R}$/MW). For T less than T_L, zero flux is applied. For T between T_L and T_H, a 3rd order smoothing polynomial is used to provide continuity of values and derivatives between 0 and the expression given above. For T greater than T_H, the above expression is used. If not provided, T_L and T_H are set to 10% below and above T_{Boil}, respectively.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output

If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Flux_Output

If specified, the average flux of the given BC/Src is calculated and stored into the global variable name specified as the *flux_output* argument.

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
power_output	"string"	–
flux_output	"string"	–
toggle	"string"	–
tboil	real	–
l	real	–
pa	real	–
mw	real	–
alpha	real	–
tl	real	-1
th	real	-1

Face_Field_Scalar

Syntax

Bc Flux For Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Face_Field_Scalar ([Power_Output = *power_output*] | [Flux_Output = *flux_output*] | [Toggle = *toggle*] | Name = *name* | [Multiplier = *multiplier*] | [Advective_Bar = *advective_bar*] | [Command_Block_Name = *command_block_name*])

Summary

FACE or EDGE Field energy flux

Description

Flux is given by:

$$M \cdot f$$

M is given by MULTIPLIER and f is a face or edge field with name NAME.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output

If specified, the total (integrated) power of the given BC/Src is

calculated and stored into the global variable name specified as the *power_output* argument.

Flux_Output

If specified, the average flux of the given BC/Src is calculated and stored into the global variable name specified as the *flux_output* argument.

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
power_output	“string”	–
flux_output	“string”	–
toggle	“string”	–
name	“string”	–
multiplier	real	1
advective_bar	“string”	–
command_block_name	“string”	–

Fortran

Syntax

Bc Flux For Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Fortran ([Power_Output = *power_output*] | [Flux_Output = *flux_output*] | [Toggle = *toggle*] | [Multiplier = *multiplier*] | Sub_Name = *sub_name* | [Real_Data = *real_data*] | [Int_Data = *int_data*] | [Resource_Name = *resource_name*] | [Data = *data*])

Summary

Fortran subroutine energy flux

Description

The following options are present in most boundary conditions/sources. The

meaning of each option is as follows:

Power_Output
If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Flux_Output
If specified, the average flux of the given BC/Src is calculated and stored into the global variable name specified as the *flux_output* argument.

Toggle
If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
power_output	“string”	–
flux_output	“string”	–
toggle	“string”	–
multiplier	real	1
sub_name	“string”	–
real_data	“string”	–
int_data	“string”	–
resource_name	“string”	–
data	“string”	–

Gaussian_Power_Weld

Syntax

Bc Flux For Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Gaussian_Power_Weld ([Power_Output = *power_output*] | [Flux_Output = *flux_output*] | [Toggle = *toggle*] | R = *r* | [Clip_Radius = *clip_radius*] | [Efficiency = *efficiency*] | R_Eff = *R_eff* | Origin_And_Normal_Function =

origin_and_normal_function | [Old_Temperature = *old_temperature*] | Path_Function = *path_function* | [Deposition = *deposition*] | [Integrated = *integrated*] | [Clip = *clip*])

Summary

Energy flux due to an incident Gaussian laser beam

Description

The coordinates of the center of the beam are provided by the X, Y, and Z components of the provided origin_and_normal_function. The direction of the beam is provided by the NX, NY, and NZ components of the origin_and_normal_function. The applied flux is given by:

$$\frac{(-\boldsymbol{n})\cdot\boldsymbol{d}}{|\boldsymbol{d}|}\frac{\alpha e R_{eff} P}{\pi R^2(1-e^{-R_{eff}})}e^{-R_{eff}\frac{r_p^2}{R^2}}$$

P is the laser power defined for the material, R is the beam radius, R_{eff} is the effective beam radius, α is the laser absorptivity defined for the material, e is the EFFICIENCY, and r_p is the shortest distance between a point on the surface and the line defined by the beam origin and normal. The term $-\boldsymbol{n}\cdot\boldsymbol{d}$ accounts for the fact that the surface may not be orthogonal to the laser. Points with values of r_p > CLIP_RADIUS will receive zero flux. If ACTIVATION is set to true, points whose temperature exceed ACTIVATION_TEMPERATURE will be flagged as activated for use with the ACTIVATION USER FUNCTION thermal conductivity model. If INTEGRATED is true, this flux will be integrated between the previous time step and the current one and the average flux applied.

See the *Directed Energy* section for the appropriate syntax to use this flux in the directed energy block.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output

If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Flux_Output

If specified, the average flux of the given BC/Src is calculated and stored into the global variable name specified as the *flux_output* argument.

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
power_output	"string"	–
flux_output	"string"	–
toggle	"string"	–
r	real	–
clip_radius	real	0
efficiency	real	1
R_eff	real	–
origin_and_normal_function	"string"	–
old_temperature	"string"	TRUE
path_function	"string"	–
deposition	"string"	FALSE
integrated	"string"	FALSE
clip	"string"	TRUE

Gaussian_Ray_Trace

Syntax

Bc Flux For Energy ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Gaussian_Ray_Trace ([Power_Output = *power_output*] | [Flux_Output = *flux_output*] | [Toggle = *toggle*] | Path_Function = *path_function* | Dir = *dir* | Power = *power* | Type = *type* | [N_Rays = *n_rays*] | [N_R = *n_r*] | [N_Theta_Initial = *n_theta_initial*] | [N_Intersections = *n_intersections*] | [Weight_Ratio = *weight_ratio*] | [Id = *id*] | [Absorption_Model = *absorption_model*] | [Parallel_Type = *parallel_type*] | R = *r* | R_Eff = *r_eff*)

Summary

Ray tracing implementation of a Gaussian Beam energy flux

Description

r - Laser Gaussian standard deviation.

r_eff - Laser exponential factor. Gaussian is given by $EXP\left(\frac{-r_eff*D^2}{r^2}\right)$, where D

is the distance between a point and the laser center.

type - Either radial discretization or Monte Carlo. Determines how beam will be discretized into rays. Radial discretization is deterministic and breaks beams into approximately equal area segments radially. Each segment generates one ray. The power of each ray varies based on which segment it is generated from. Monte Carlo is stochastic and will generate a user-specified number of rays distributed spatially according to the laser's shape function, with each ray having equal power of power/n_rays.

n_r - Number of segments in radial direction for radial discretization.

n_theta_initial - Number of segments in theta direction in the smallest radial segment. The number of theta segments will grow as radial direction increases so segments have approximately the same area. Total number of rays will be n_r*n_r*n_theta_initial.

n_rays - Number of rays to generate for Monte Carlo discretization.

path_function - Function giving x, y, z coordinates of laser center as function of time.

dir - Function giving x,y,z components of direction that rays will travel initially from laser center defined by path_function.

power - Function giving total laser power.

n_intersections - Maximum number of times to intersect a ray with mesh elements due to reflections (default: 10).

weight_ratio - Fraction of initial ray power below which rays will be considered spent and no longer be traced (default: 0.001).

absorption_model - Either Specular, to use a surface emissivity value to determine fraction of energy absorbed and reflected with each ray intersection or Fresnel to use Fresnel model dependent on ray angle of incidence.

parallel_type - Either ray_parallel, to divide rays between processors with a globally shared mesh part or scene_parallel, to divide mesh part between processors with a globally shared set of rays.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output
: If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Flux_Output
: If specified, the average flux of the given BC/Src is calculated and

stored into the global variable name specified as the *flux_output* argument.

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
power_output	“string”	–
flux_output	“string”	–
toggle	“string”	–
path_function	“string”	–
dir	“string”	–
power	“string”	–
type	“string”	–
n_rays	integer	0
n_r	integer	0
n_theta_initial	integer	0
n_intersections	integer	10
weight_ratio	real	0.001
id	“string”	–
absorption_model	“string”	SPECULAR
parallel_type	“string”	RAY_PARALLEL
r	real	–
r_eff	real	–

Generalized_Nat_Conv

Syntax

Bc Flux For Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Generalized_Nat_Conv ([Power_Output = *power_output*] | [Flux_Output = *flux_output*] | [Toggle = *toggle*] | [Multiplier = *multiplier*] | [H = *h*] | [T_Ref = *t_ref*])

Summary

Generalized natural convection energy flux

Description

Flux is given by:

$$F = Mh\left(T - T_{ref}\right)$$

M is given by MULTIPLIER, T_{ref} and h are taken from the BC REFERENCE TEMPERATURE and HEAT TRANSFER COEFFICIENT expressions.

The values for h and T_{ref} can also be supplied in-line as either constants or string functions.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output

If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Flux_Output

If specified, the average flux of the given BC/Src is calculated and stored into the global variable name specified as the *flux_output* argument.

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
power_output	“string”	–
flux_output	“string”	–
toggle	“string”	–
multiplier	real	1
h	“string”	–
t_ref	“string”	–

Generalized_Rad

Syntax

Bc Flux For Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Generalized_Rad ([Power_Output = *power_output*] | [Flux_Output = *flux_output*] | [Toggle = *toggle*] | [Multiplier = *multiplier*] | [T_Ref = *t_ref*])

Summary

Radiative energy flux, with values for emissivity, form factor and irradiation taken from expressions

Description

Flux is:

$$M\epsilon f\left(\sigma T^4 - I\right)$$

M is MULTIPLIER. σ is the Stefan-Boltzmann constant. ϵ and F are expressions for emissivity and form factor. T is surface temperature. I is the expression for irradiation. If I does not already exist, it will be constructed as σT_{ref}^4, where T_{ref} is either provided by the BC Rad Reference Temperature expression or the inline T_{ref} parameter (constant or string function).

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output

If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Flux_Output

If specified, the average flux of the given BC/Src is calculated and stored into the global variable name specified as the *flux_output* argument.

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
power_output	“string”	–
flux_output	“string”	–
toggle	“string”	–
multiplier	real	1
t_ref	“string”	–

Global

Syntax

Bc Flux For Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Power_Output = *power_output*] | [Flux_Output = *flux_output*] | [Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output

If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Flux_Output

If specified, the average flux of the given BC/Src is calculated and stored into the global variable name specified as the *flux_output* argument.

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
power_output	"string"	–
flux_output	"string"	–
toggle	"string"	–
global_name	"string"	–
component	integer	0

Kuntz

Syntax

Bc Flux For Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Kuntz ([Power_Output = *power_output*] | [Flux_Output = *flux_output*] | [Toggle = *toggle*] | [Multiplier = *multiplier*] | File = *file* | Units = *units*)

Summary

Interpolated value from tabulated data for the energy flux. FILE provides the filename for a table which specifies a flux value given X, Y, and Z coordinates and a temperature. UNITS specifies the units the table is in - ENGLISH ($\frac{BTU}{sft^2}$) or METRIC($\frac{W}{m^2}$)

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output

If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Flux_Output

If specified, the average flux of the given BC/Src is calculated and stored into the global variable name specified as the *flux_output* argument.

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
power_output	"string"	–
flux_output	"string"	–
toggle	"string"	–
multiplier	real	1
file	"string"	–
units	"string"	–

Lens

Syntax

Bc Flux For Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Lens ([Power_Output = *power_output*] | [Flux_Output = *flux_output*] | [Toggle = *toggle*] | Src_X = *src_x* | Dir_X = *dir_x* | Vel_X = *vel_x* | Src_Y = *src_y* | Dir_Y = *dir_y* | Vel_Y = *vel_y* | [Src_Z = *src_z*] | [Dir_Z = *dir_z*] | [Vel_Z = *vel_z*] | Massflowrate = *massFlowRate* | T_Liq = *T_liq* | T_Solid = *T_solid* | [T_On = *t_on*] | [T_Off = *t_off*] | Masssourceradius = *massSourceRadius* | Particletemperature = *particleTemperature* | Particledensity = *particleDensity*)

Summary

Energy flux for modeling the LENS process

Description

This boundary condition imposes an energy flux due to an incident mass source representing a powder particle stream due to the LENS process. The source is at the coordinates

$$\boldsymbol{s} = \boldsymbol{x}_s + \boldsymbol{v}_s(t - T_ON)$$

where $\boldsymbol{x}_s$ is provided by the SRC vector coordinates and $\boldsymbol{v}_s$ is provided by the VEL vector coordinates. The source is directed from by the DIR vector (). The

point-wise energy flux is computed using the equation below:

$$\frac{(-\boldsymbol{n})\cdot\boldsymbol{d}}{|\boldsymbol{d}|}\left(\rho c_p T\frac{\dot{m}}{\rho_p \pi R^2}-\frac{\dot{m}c_p T_p}{\pi R^2}\right)$$

ρ, c_p, and T are the density, specific heat, and temperature of the surface. R, $\dot{m}$, ρ_p, and T_p are given by the massSourceRadius, massFlowRate, particleDensity, and particleTemperature parameters. The term $-\boldsymbol{n}\cdot\boldsymbol{d}$ accounts for the fact that the surface may not be orthogonal to the source. For points where the minimum distance between the point and the line defined by and is greater than R, the flux will be set to zero. Additionally, if T is less than (T_liq+T_solid)/2, the flux will be set to zero. For times less than T_ON or greater than T_OFF, the flux will also be set to zero.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output
: If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Flux_Output
: If specified, the average flux of the given BC/Src is calculated and stored into the global variable name specified as the *flux_output* argument.

Toggle
: If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
power_output	“string”	–
flux_output	“string”	–
toggle	“string”	–
src_x	real	–
dir_x	real	–
vel_x	real	–
src_y	real	–
dir_y	real	–
vel_y	real	–
src_z	real	0
dir_z	real	0
vel_z	real	0
massFlowRate	real	–
T_liq	real	–
T_solid	real	–
t_on	real	0
t_off	real	1.798e+308
massSourceRadius	real	–
particleTemperature	real	–
particleDensity	real	–

Nat_Conv

Syntax

Bc Flux For Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nat_Conv ([Power_Output = *power_output*] | [Flux_Output = *flux_output*] | [Toggle = *toggle*] | H = *h* | T_Ref = *t_ref*)

Summary

Convective energy flux defined with constants specified for heat transfer coefficient and reference temperature.

Description

Flux is given by:

$$F = h\left(T - T_{ref}\right)$$

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output

If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Flux_Output

If specified, the average flux of the given BC/Src is calculated and stored into the global variable name specified as the *flux_output* argument.

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
power_output	"string"	–
flux_output	"string"	–
toggle	"string"	–
h	real	–
t_ref	real	–

Nd_Table

Syntax

Bc Flux For Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Power_Output = *power_output*] | [Flux_Output = *flux_output*] | [Toggle =

toggle] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table.

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output

If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Flux_Output

If specified, the average flux of the given BC/Src is calculated and stored into the global variable name specified as the *flux_output* argument.

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
power_output	“string”	–
flux_output	“string”	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Pid_Controlled

Syntax

Bc Flux For Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Pid_Controlled ([Power_Output = *power_output*] | [Flux_Output = *flux_output*] | [Toggle = *toggle*] | Max_Output = *Max_Output* | [P = *p*] | [I = *i*] | [D = *d*] | [T_I = *t_i*] | [T_D = *t_d*] | [Band = *Band*] | [Filter_Tau = *Filter_Tau*] | [Filter_Number = *Filter_Number*] | Setpoint_Function_Name = *Setpoint_Function_Name* | Control_Variable = *Control_Variable* | [Enable_Clamping = *enable_clamping*])

Summary

This flux term imposes an energy flux using a PID controller

Description

The controller setpoint vs. time is defined in a user function, and the controller feedback point uses a global variable (e.g. a data probe or reduction postprocessor).

The derivative term uses a filtered error value, using a low pass filter whose time constant is specified by the user. Like a heater, this is only capable of providing a positive influx. Even if the controller output is negative, the applied energy source will be truncated at 0. Likewise, the applied energy source cannot exceed the specified max source.

The nominal scaled PID controller output (between 0 and 1) is

$$u(t) = K_p e(t) + K_i \int_0^t e(s)ds + K_d \frac{\partial e_f}{\partial t}$$

where it is important to note that the K_p coefficient does not scale the others unless you have set it using the Band argument. In some PID implementations, the K_i and K_d coefficients are internally multiplied by K_p so care must be taken to provide the correct coefficients. This output $u(t)$ which is from 0 to 1 is scaled by the specified Max_Output value to produce the applied energy source. This means you **should not** include the output magnitude scaling in your P, I, or D parameters. If this controller is meant to recreate a controller that commands power directly, one must normalize by the max power.

When a derivative gain is provided, the derivative action is calculated using a filtered error to avoid derivative kick. The `Filter_Tau` argument can be used to specify the first-order derivative filter time constant τ, with a default value of 1 otherwise. Alternatively, one can provide the `Filter_Number` as argument $\tau = \frac{1}{N}$ where

$$K_d \frac{N}{1 + N\frac{1}{s}} = K_d \frac{1}{\tau + \frac{1}{s}} = \frac{K_d s}{\tau s + 1}$$

The P, I, and D coefficients can be specified in a number of ways:

- Just give a 'Band' argument. This calculates $P = 1/Band$ and uses the defaults of $I = 0.01P$ and $D = 0.0$
- Give a 'Band' argument and non-defaults for I and/or D. This still multiplies whatever you give for I and D by P.
- Give a 'P' argument instead of 'Band'. NOTE: in this mode the I and D values are not scaled by P.

If you provide both P and Band, P is used, the Band argument is ignored, and I and D are not scaled.

Alternatively, the integral and/or derivative gains can be provided using the integral time T_i (`T_i` argument) and derivative time T_d (`T_d` argument) so that $I = P/T_i$ and $D = P * T_d$ (regardless if P was set via band arg or not).

If `ENABLE_CLAMPING` is set to true, a clamping anti-windup algorithm is applied to the integral action of the controller.

The default values for I and D are 0.01 and 0 if not provided. The setpoint, controller output, error, filtered error, and filtered error time derivative are all output to automatically created global variables for diagnostics and postprocessing.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output
: If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Flux_Output
: If specified, the average flux of the given BC/Src is calculated and stored into the global variable name specified as the *flux_output* argument.

Toggle
: If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
power_output	"string"	–
flux_output	"string"	–
toggle	"string"	–
Max_Output	real	–
p	real	0
i	real	0
d	real	0
t_i	real	0
t_d	real	0
Band	real	0
Filter_Tau	real	1
Filter_Number	real	1
Setpoint_Function_Name	"string"	–
Control_Variable	"string"	–
enable_clamping	integer	–

Pid_Controlled_Bidirectional

Syntax

Bc Flux For Energy ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Pid_Controlled_Bidirectional ([Power_Output = *power_output*] | [Flux_Output = *flux_output*] | [Toggle = *toggle*] | Max_Cooling_Power = *Max_Cooling_Power* | Max_Heating_Power = *Max_Heating_Power* | [P = *p*] | [I = *i*] | [D = *d*] | [T_I = *t_i*] | [T_D = *t_d*] | [Band = *Band*] | [Filter_Tau = *Filter_Tau*] | [Filter_Number = *Filter_Number*] | Setpoint_Function_Name = *Setpoint_Function_Name* | Control_Variable = *Control_Variable* | [Enable_Clamping = *enable_clamping*])

Summary

This is similar to PID_Controlled, however, the flux is applied to cool and heat the volume instead of just heating it. Max_Cooling_Power and Max_Heating_Power should be positive values. See description for PID_CONTROLLED for more details

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output

If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Flux_Output

If specified, the average flux of the given BC/Src is calculated and stored into the global variable name specified as the *flux_output* argument.

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
power_output	"string"	–
flux_output	"string"	–
toggle	"string"	–
Max_Cooling_Power	real	–
Max_Heating_Power	real	–
p	real	0
i	real	0
d	real	0
t_i	real	0
t_d	real	0
Band	real	0
Filter_Tau	real	1
Filter_Number	real	1
Setpoint_Function_Name	"string"	–
Control_Variable	"string"	–
enable_clamping	integer	–

Pid_Controlled_Cooler

Syntax

Bc Flux For Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Pid_Controlled_Cooler ([Power_Output = *power_output*] | [Flux_Output = *flux_output*] | [Toggle = *toggle*] | [P = *p*] | [I = *i*] | [D = *d*] | [T_I = *t_i*] | [T_D = *t_d*] | [Band = *Band*] | [Filter_Tau = *Filter_Tau*] | [Filter_Number = *Filter_Number*] | Setpoint_Function_Name = *Setpoint_Function_Name* | Control_Variable = *Control_Variable* | [Enable_Clamping = *enable_clamping*] | Max_Cooling_Power = *Max_Cooling_Power*)

Summary

This is similar to PID_Controlled, however, the flux is applied to cool the volume instead of heat it. Max_Cooling_Power should be a positive value. See description for PID_CONTROLLED for more details

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output

If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Flux_Output

If specified, the average flux of the given BC/Src is calculated and stored into the global variable name specified as the *flux_output* argument.

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
power_output	"string"	–
flux_output	"string"	–
toggle	"string"	–
p	real	0
i	real	0
d	real	0
t_i	real	0
t_d	real	0
Band	real	0
Filter_Tau	real	1
Filter_Number	real	1
Setpoint_Function_Name	"string"	–
Control_Variable	"string"	–
enable_clamping	integer	–
Max_Cooling_Power	real	–

Polynomial

Syntax

Bc Flux For Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Power_Output = *power_output*] | [Flux_Output = *flux_output*] | [Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The flux, F, is calculated as $F = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The

meaning of each option is as follows:

Power_Output
: If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Flux_Output
: If specified, the average flux of the given BC/Src is calculated and stored into the global variable name specified as the *flux_output* argument.

Toggle
: If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
power_output	“string”	–
flux_output	“string”	–
toggle	“string”	–
variable	“string”	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Rad

Syntax

Bc Flux For Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Rad ([Power_Output = *power_output*] | [Flux_Output = *flux_output*] | [Toggle = *toggle*] | Crad = *crad* | T_Ref = *t_ref*)

Summary

Radiative energy flux defined with constant models for emissivity and reference temperature.

Description

The applied flux is given by:

$$c_{rad}\left(T^4 - T_{ref}^4\right)$$

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output

If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Flux_Output

If specified, the average flux of the given BC/Src is calculated and stored into the global variable name specified as the *flux_output* argument.

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
power_output	"string"	–
flux_output	"string"	–
toggle	"string"	–
crad	real	–
t_ref	real	–

Ray_Trace

Syntax

Bc Flux For Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Ray_Trace ([Power_Output = *power_output*] | [Flux_Output = *flux_output*] | [Toggle = *toggle*] | Ray_Source = *ray_source* | [Emission = *emission*])

Summary

Apply a ray tracing energy flux

Description

Adds an energy flux based on the intersection of rays defined in a ray tracing source block for the equation system with name given by the RAY_SOURCE parameter with the surface. The optional EMISSION parameter also imposes a σT^4 blackbody radiative emission term

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output

If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Flux_Output

If specified, the average flux of the given BC/Src is calculated and stored into the global variable name specified as the *flux_output* argument.

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
power_output	"string"	–
flux_output	"string"	–
toggle	"string"	–
ray_source	"string"	–
emission	"string"	FALSE

Rte_Field

Syntax

Bc Flux For Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Rte_Field ([Power_Output = *power_output*] | [Flux_Output = *flux_output*] | [Toggle = *toggle*])

Summary

RTE energy flux from a field

Description

Flux is given by -f, where f is the field named 'rte_radiative_flux'

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output

If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Flux_Output

If specified, the average flux of the given BC/Src is calculated and

stored into the global variable name specified as the *flux_output* argument.

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
power_output	"string"	–
flux_output	"string"	–
toggle	"string"	–

Rte_Sp

Syntax

Bc Flux For Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Rte_Sp ([Power_Output = *power_output*] | [Flux_Output = *flux_output*] | [Toggle = *toggle*] | Order = *order*)

Summary

RTE flux for energy equation with Simplified Spherical Harmonic RTE model with order ORDER

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output

If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Flux_Output

If specified, the average flux of the given BC/Src is calculated and stored into the global variable name specified as the *flux_output*

argument.

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
power_output	"string"	–
flux_output	"string"	–
toggle	"string"	–
order	integer	–

Scalar_String_Function

Syntax

Bc Flux For Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Power_Output = *power_output*] | [Flux_Output = *flux_output*] | [Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output

If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Flux_Output

If specified, the average flux of the given BC/Src is calculated and stored into the global variable name specified as the *flux_output* argument.

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
power_output	"string"	–
flux_output	"string"	–
toggle	"string"	–
f	"string"	–

Scaled_Nat_Conv

Syntax

Bc Flux For Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scaled_Nat_Conv ([Power_Output = *power_output*] | [Flux_Output = *flux_output*] | [Toggle = *toggle*] | [Multiplier = *multiplier*] | Scaling_Field = *scaling_field* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Scaled natural convection energy flux

Description

Flux is given by:

$$(M)(SC)(GV)(f)fcn(t)h\left(T-T_{ref}\right)$$

M is given by MULTIPLIER, SC by SCALING, GV by the value of the global variable with name GLOBAL_VAR, fcn is a function with name TIME_FUNCTION, and f is a field with name SCALING_FIELD. h and T_{ref} are taken from the HEAT TRANSFER COEFFICIENT and BC REFERENCE TEMPERATURE expressions.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output

If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Flux_Output

If specified, the average flux of the given BC/Src is calculated and stored into the global variable name specified as the *flux_output* argument.

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
power_output	“string”	–
flux_output	“string”	–
toggle	“string”	–
multiplier	real	1
scaling_field	“string”	–
scaling	real	1
global_var	“string”	–
time_function	“string”	–

Sharp_Power_Weld

Syntax

Bc Flux For Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Sharp_Power_Weld ([Power_Output = *power_output*] | [Flux_Output = *flux_output*] | [Toggle = *toggle*] | R = *r* | [Clip_Radius = *clip_radius*] | [Efficiency = *efficiency*] | Origin_And_Normal_Function = *origin_and_normal_function* | [Old_Temperature = *old_temperature*] | Path_Function = *path_function* | [Deposition = *deposition*] | [Integrated = *integrated*] | [Clip = *clip*])

Summary

Energy flux due to an incident 'top hat' beam

Description

The coordinates of the center of the beam are provided by the X, Y, and Z components of the provided origin_and_normal_function. The direction of the beam is provided by the NX, NY, and NZ components of the origin_and_normal_function. The applied flux is given by:

$$\frac{(-\boldsymbol{n}) \cdot \boldsymbol{d}}{|\boldsymbol{d}|} \frac{\alpha e P}{\pi R^2}$$

P is the laser power defined for the material, R is the beam radius, α is the laser absorptivity defined for the material, e is the EFFICIENCY, and the term $-\boldsymbol{n} \cdot \boldsymbol{d}$ accounts for the fact that the surface may not be orthogonal to the laser. Points with values of r_p > CLIP_RADIUS will receive zero flux. r_p is the shortest distance between a point on the surface and the line defined by the beam origin and normal. If ACTIVATION is set to true, points whose temperature exceed ACTIVATION_TEMPERATURE will be flagged as activated for use with the ACTIVATION USER FUNCTION thermal conductivity model. If INTEGRATED is true, this flux will be integrated between the previous time step and the current one and the average flux applied.

See the *Directed Energy* section for the appropriate syntax to use this flux in the directed energy block.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output

If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Flux_Output

If specified, the average flux of the given BC/Src is calculated and stored into the global variable name specified as the *flux_output* argument.

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
power_output	"string"	–
flux_output	"string"	–
toggle	"string"	–
r	real	–
clip_radius	real	0
efficiency	real	1
origin_and_normal_function	"string"	–
old_temperature	"string"	TRUE
path_function	"string"	–
deposition	"string"	FALSE
integrated	"string"	FALSE
clip	"string"	TRUE

Sharp_Ray_Trace

Syntax

Bc Flux For Energy ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Sharp_Ray_Trace ([Power_Output = *power_output*] | [Flux_Output = *flux_output*] | [Toggle = *toggle*] | Path_Function = *path_function* | Dir = *dir* | Power = *power* | Type = *type* | [N_Rays = *n_rays*] | [N_R = *n_r*] | [N_Theta_Initial = *n_theta_initial*] | [N_Intersections = *n_intersections*] | [Weight_Ratio = *weight_ratio*] | [Id = *id*] | [Absorption_Model = *absorption_model*] | [Parallel_Type = *parallel_type*] | R = *r*)

Summary

Ray tracing implementation of a sharp circular beam energy flux

Description

r - Laser spot radius.

type - Either radial discretization or Monte Carlo. Determines how beam will be discretized into rays. Radial discretization is deterministic and breaks beams into approximately equal area segments radially. Each segment generates one

ray. The power of each ray varies based on which segment it is generated from. Monte Carlo is stochastic and will generate a user-specified number of rays distributed spatially according to the laser's shape function, with each ray having equal power of power/n_rays.

n_r - Number of segments in radial direction for radial discretization.

n_theta_initial - Number of segments in theta direction in the smallest radial segment. The number of theta segments will grow as radial direction increases so segments have approximately the same area. Total number of rays will be n_r*n_r*n_theta_initial.

n_rays - Number of rays to generate for Monte Carlo discretization.

path_function - Function giving x, y, z coordinates of laser center as function of time.

dir - Function giving x,y,z components of direction that rays will travel initially from laser center defined by path_function.

power - Function giving total laser power.

n_intersections - Maximum number of times to intersect a ray with mesh elements due to reflections (default: 10).

weight_ratio - Fraction of initial ray power below which rays will be considered spent and no longer be traced (default: 0.001).

absorption_model - Either Specular, to use a surface emissivity value to determine fraction of energy absorbed and reflected with each ray intersection or Fresnel to use Fresnel model dependent on ray angle of incidence.

parallel_type - Either ray_parallel, to divide rays between processors with a globally shared mesh part or scene_parallel, to divide mesh part between processors with a globally shared set of rays.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output
: If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Flux_Output
: If specified, the average flux of the given BC/Src is calculated and stored into the global variable name specified as the *flux_output* argument.

Toggle
: If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
power_output	“string”	–
flux_output	“string”	–
toggle	“string”	–
path_function	“string”	–
dir	“string”	–
power	“string”	–
type	“string”	–
n_rays	integer	0
n_r	integer	0
n_theta_initial	integer	0
n_intersections	integer	10
weight_ratio	real	0.001
id	“string”	–
absorption_model	“string”	SPECULAR
parallel_type	“string”	RAY_PARALLEL
r	real	–

Thermal_Latent_Heat

Syntax

Bc Flux For Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Thermal_Latent_Heat ([Power_Output = *power_output*] | [Flux_Output = *flux_output*] | [Toggle = *toggle*])

Summary

Thermal latent heat energy flux

Description

Flux is given by:

$$H_v v_e$$

H_v is the expression for latent heat of vaporization and v_e is the local interface speed

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output
If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Flux_Output
If specified, the average flux of the given BC/Src is calculated and stored into the global variable name specified as the *flux_output* argument.

Toggle
If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
power_output	"string"	–
flux_output	"string"	–
toggle	"string"	–

User_Field

Syntax

Bc Flux For Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Power_Output = *power_output*] | [Flux_Output = *flux_output*] | [Toggle = *toggle*] | [Multiplier = *multiplier*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

User field energy flux

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output

If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Flux_Output

If specified, the average flux of the given BC/Src is calculated and stored into the global variable name specified as the *flux_output* argument.

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
power_output	"string"	–
flux_output	"string"	–
toggle	"string"	–
multiplier	real	1
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc Flux For Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Power_Output = *power_output*] | [Flux_Output = *flux_output*] | [Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier =

multiplier])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output

If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Flux_Output

If specified, the average flux of the given BC/Src is calculated and stored into the global variable name specified as the *flux_output* argument.

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
power_output	“string”	–
flux_output	“string”	–
toggle	“string”	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc Flux For Energy ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Power_Output = *power_output*] | [Flux_Output = *flux_output*] | [Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output

If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Flux_Output

If specified, the average flux of the given BC/Src is calculated and stored into the global variable name specified as the *flux_output* argument.

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
power_output	“string”	–
flux_output	“string”	–
toggle	“string”	–
Name	“string”	–
plugin_parameters	“string”	–

User_Vector_Field

Syntax

Bc Flux For Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Vector_Field ([Power_Output = *power_output*] | [Flux_Output = *flux_output*] | [Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*] | [Multiplier = *multiplier*])

Summary

User vector field energy flux

Description

Flux is given by:

$$(M)(SC)(GV)fcn(t)\boldsymbol{f}\cdot\boldsymbol{n}$$

M is given by MULTIPLIER, SC by SCALING, GV by the value of the global variable with name GLOBAL_VAR, fcn is a function with name TIME_FUNCTION, f is a field with name NAME, and n is the surface normal

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output

If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Flux_Output

If specified, the average flux of the given BC/Src is calculated and stored into the global variable name specified as the *flux_output* argument.

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
power_output	"string"	–
flux_output	"string"	–
toggle	"string"	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–
multiplier	real	1

User_Vector_Field_Influx

Syntax

Bc Flux For Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Vector_Field_Influx ([Power_Output = *power_output*] | [Flux_Output = *flux_output*] | [Toggle = *toggle*] | [Multiplier = *multiplier*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

User vector field influx energy flux

Description

Flux is given by:

$$(M)(SC)(GV)fcn(t)\boldsymbol{f}\cdot\boldsymbol{n}$$

M is given by MULTIPLIER, SC by SCALING, GV by the value of the global variable with name GLOBAL_VAR, fcn is a function with name TIME_FUNCTION, f is a field with name NAME, and n is the surface normal. Flux will be capped at zero (inflow only)

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output

If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Flux_Output

If specified, the average flux of the given BC/Src is calculated and stored into the global variable name specified as the *flux_output* argument.

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
power_output	“string”	–
flux_output	“string”	–
toggle	“string”	–
multiplier	real	1
name	“string”	–
scaling	real	1
global_var	“string”	–
time_function	“string”	–

Vapor_Cooling

Syntax

Bc Flux For Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Vapor_Cooling ([Power_Output = *power_output*] | [Flux_Output = *flux_output*] | [Toggle = *toggle*] | Tboil = *tboil*)

Summary

Empirical relation for cooling due to vaporization of stainless steel.

Description

Flux is given by:

$$814373(T - T_{boil}) - 2248.31(T - T_{boil})^2 + 27.1683(T - T_{boil})^3$$

For $T - T_{boil}$ less than 170 and:

$$-310360000 + 3.272400(T - T_{boil}) - 1808.4(T - T_{boil})^2 + 2.7284(T - T_{boil})^3$$

For $T - T_{boil}$ greater than 170. For $T - T_{boil}$ less than zero, no flux is applied

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output

If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Flux_Output

If specified, the average flux of the given BC/Src is calculated and stored into the global variable name specified as the *flux_output* argument.

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
power_output	“string”	–
flux_output	“string”	–
toggle	“string”	–
tboil	real	–

Enthalpy

Constant

Syntax

Bc Flux For Enthalpy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | Value = *value*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
value	real	–

Convective_Outflow

Syntax

Bc Flux For Enthalpy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Convective_Outflow [Toggle = *toggle*]

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–

Exponential

Syntax

Bc Flux For Enthalpy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Generalized_Nat_Conv

Syntax

Bc Flux For Enthalpy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Generalized_Nat_Conv ([Toggle = *toggle*] | [Multiplier = *multiplier*])

Summary

Generalized natural convection enthalpy flux

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
multiplier	real	1

Generalized_Rad

Syntax

Bc Flux For Enthalpy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Generalized_Rad ([Toggle = *toggle*] | [Multiplier = *multiplier*] | [T_Ref = *t_ref*])

Summary

Radiative energy flux, with values for emissivity, form factor and irradiation taken from expressions

Description

Flux is:

$$M\epsilon f\left(\sigma T^4 - I\right)$$

M is MULTIPLIER. σ is the Stefan-Boltzmann constant. ϵ and F are expressions for emissivity and form factor. T is surface temperature. I is the expression for irradiation. If I does not already exist, it will be constructed as σT_{ref}^4, where T_{ref} is either provided by the BC Rad Reference Temperature expression or the inline T_{ref} parameter (constant or string function).

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
multiplier	real	1
t_ref	"string"	–

Global

Syntax

Bc Flux For Enthalpy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
global_name	“string”	–
component	integer	0

Nat_Conv

Syntax

Bc Flux For Enthalpy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nat_Conv ([Toggle = *toggle*] | H = *h* | T_Ref = *t_ref*)

Summary

Natural convection BC for enthalpy flux

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
h	real	–
t_ref	real	–

Nd_Table

Syntax

Bc Flux For Enthalpy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc Flux For Enthalpy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
variable	“string”	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Rad

Syntax

Bc Flux For Enthalpy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Rad ([Toggle = *toggle*] | Crad = *crad* | T_Ref = *t_ref*)

Summary

Radiative energy flux defined with constant models for emissivity and reference temperature.

Description

The applied flux is given by:

$$c_{rad}\left(T^4 - T_{ref}^4\right)$$

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
crad	real	–
t_ref	real	–

Scalar_String_Function

Syntax

Bc Flux For Enthalpy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f	"string"	–

User_Field

Syntax

Bc Flux For Enthalpy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc Flux For Enthalpy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc Flux For Enthalpy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

Level_Set

Constant

Syntax

Bc Flux For Level_Set ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | *{flux | value}* = *flux*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
flux	real	–

Exponential

Syntax

Bc Flux For Level_Set ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Flux For Level_Set ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
global_name	"string"	–
component	integer	0

Inflow_Outflow

Syntax

Bc Flux For Level_Set ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Inflow_Outflow [Toggle = *toggle*]

Summary

This model has been deprecated and removed from the code.

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–

Nd_Table

Syntax

Bc Flux For Level_Set ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc Flux For Level_Set ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
variable	“string”	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Flux For Level_Set ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
f	“string”	–

User_Field

Syntax

Bc Flux For Level_Set ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc Flux For Level_Set ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc Flux For Level_Set ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

Lumped_Div_Projection

Constant

Syntax

Bc Flux For Lumped_Div_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | [Valuex = *valuex*] | [Valuey = *valuey*] | [Valuez = *valuez*])

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
valuex	real	0
valuey	real	0
valuez	real	0

Vector_String_Function

Syntax

Bc Flux For Lumped_Div_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Vector_String_Function ([Toggle = *toggle*] | F_X = *f_x* | F_Y = *f_y* | [F_Z = *f_z*])

Summary

Value from a string function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f_x	"string"	–
f_y	"string"	–
f_z	"string"	–

Lumped_Div_Projection_X

Constant

Syntax

Bc Flux For Lumped_Div_Projection_X ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | Value = *value*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
value	real	–

Exponential

Syntax

Bc Flux For Lumped_Div_Projection_X ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Flux For Lumped_Div_Projection_X ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Bc Flux For Lumped_Div_Projection_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc Flux For Lumped_Div_Projection_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Flux For Lumped_Div_Projection_X ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f	"string"	–

User_Field

Syntax

Bc Flux For Lumped_Div_Projection_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc Flux For Lumped_Div_Projection_X ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc Flux For Lumped_Div_Projection_X ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

Lumped_Div_Projection_Y

Constant

Syntax

Bc Flux For Lumped_Div_Projection_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | Value = *value*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
value	real	–

Exponential

Syntax

Bc Flux For Lumped_Div_Projection_Y ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Flux For Lumped_Div_Projection_Y ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Bc Flux For Lumped_Div_Projection_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc Flux For Lumped_Div_Projection_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Flux For Lumped_Div_Projection_Y ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f	"string"	–

User_Field

Syntax

Bc Flux For Lumped_Div_Projection_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc Flux For Lumped_Div_Projection_Y ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc Flux For Lumped_Div_Projection_Y ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

Lumped_Div_Projection_Z

Constant

Syntax

Bc Flux For Lumped_Div_Projection_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | Value = *value*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
value	real	–

Exponential

Syntax

Bc Flux For Lumped_Div_Projection_Z ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Flux For Lumped_Div_Projection_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Bc Flux For Lumped_Div_Projection_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc Flux For Lumped_Div_Projection_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Flux For Lumped_Div_Projection_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f	"string"	–

User_Field

Syntax

Bc Flux For Lumped_Div_Projection_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc Flux For Lumped_Div_Projection_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc Flux For Lumped_Div_Projection_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

Mass_Balance

Co2_Convective_Outflow

Syntax

Bc Flux For Mass_Balance ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Co2_Convective_Outflow ([Toggle = *toggle*] | [Ref_Frac = *ref_frac*])

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
ref_frac	real	0

Constant

Syntax

Bc Flux For Mass_Balance ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | Value = *value*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
value	real	–

Constant_Velocity

Syntax

Bc Flux For Mass_Balance ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant_Velocity ([Toggle = *toggle*] | Mult = *Mult*)

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is

active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Mult	real	–

Convective_Outflow

Syntax

Bc Flux For Mass_Balance ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Convective_Outflow ([Toggle = *toggle*] | [Ref_Frac = *ref_frac*])

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
ref_frac	real	0

Exponential

Syntax

Bc Flux For Mass_Balance ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
variable	“string”	–
constant	real	0
multiplier	real	1
exponent	real	–

Fluid_Robin_Coupled_One_Region

Syntax

Bc Flux For Mass_Balance ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Fluid_Robin_Coupled_One_Region ([Toggle = *toggle*] | [Coeff_Scaling = *coeff_scaling*])

Summary

Boundary condition for coupling of continuity to a free fluid in the same region.

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
coeff_scaling	real	0.001

Global

Syntax

Bc Flux For Mass_Balance ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The

meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
global_name	"string"	–
component	integer	0

Mass_Fluid_Robin_Coupled

Syntax

Bc Flux For Mass_Balance ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Mass_Fluid_Robin_Coupled ([Toggle = *toggle*] | [Coeff_Scaling = *coeff_scaling*] | P_Field = *P_Field* | Massflux_Field = *MassFlux_Field*)

Summary

Boundary condition for loose coupling of continuity to a free fluid region

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
coeff_scaling	real	0.001
P_Field	"string"	–
MassFlux_Field	"string"	–

Mass_Fraction_Fluid_Robin_Coupled

Syntax

Bc Flux For Mass_Balance ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Mass_Fraction_Fluid_Robin_Coupled ([Toggle = *toggle*] | Mass_Fraction_Field = *mass_fraction_field* | Mass_Fraction_Diffusive_Flux_Field = *mass_fraction_diffusive_flux_field* | Coeff_Field = *coeff_field*)

Summary

Boundary condition for loose coupling of scalars to a free fluid region

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
mass_fraction_field	"string"	–
mass_fraction_diffusive_flux_field	"string"	–
coeff_field	"string"	–

Mass_Fraction_Fluid_Robin_Coupled_One_Region

Syntax

Bc Flux For Mass_Balance ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Mass_Fraction_Fluid_Robin_Coupled_One_Region [Toggle = *toggle*]

Summary

Boundary condition for coupling of scalars to a free fluid in the same region.

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–

Mass_Open

Syntax

Bc Flux For Mass_Balance ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Mass_Open ([Toggle = *toggle*] | [Pressure = *pressure*])

Summary

A weakly enforced pressure condition for open flow from a porous flow domain

Description

This model weakly enforces a pressure condition on the boundary of a porous domain, either equal to the specified constant `PRESSURE`, or set using the `BOUNDARY_PRESSURE` model from the surface material.

Note: This should only be used with porous flow equations.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
pressure	real	0

Mesh_Motion

Syntax

Bc Flux For Mass_Balance ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Mesh_Motion [Toggle = *toggle*]

Summary

Moving boundary condition for Arbitrary Lagrangian-Eulerian (ALE) mesh motion in relative flux form. Accounts for the boundary advective flux due to general mesh motion.

Description

Applies the following boundary flux

$$-\int q\hat{\mathbf{u}}\cdot\mathbf{n}dS,$$

where $\hat{u}$ is the mesh velocity and where the integrated scalar q depends on the equation. For the gas phase `porous_enthalpy` equation with a `transported enthalpy = porous`, $q = \phi\rho_g h_g$. Similarly, for the gas phase `mass_balance` equation, $q = \phi\rho_g$ is the bulk mass density.

This flux boundary condition is automatically applied to the mass balance and porous enthalpy equations when the `use relative mass flux` porous flow option is turned on.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–

Moving_Boundary

Syntax

Bc Flux For Mass_Balance ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Moving_Boundary [Toggle = *toggle*]

Summary

No penetration moving boundary condition, valid for Lagrangian mesh motion in non-relative flux form or relative flux form.

Description

Applies the following boundary flux

$$\int q\hat{\mathbf{u}} \cdot \mathbf{n} dS,$$

where $\hat{u}$ is the mesh velocity and where the integrated scalar q depends on the equation. If we consider the gas phase `porous_enthalpy` equation with a `transported enthalpy = porous`, then $q = \phi\rho_g h_g$. Similarly, for the gas phase `mass_balance` equation, $q = \phi\rho_g$ is the bulk mass density.

If we consider the gas phase `mass_balance` equation, with a gas phase porous velocity relative to the moving solid phase, then physically this boundary condition is derived from applying a no flux condition at moving boundary i.e.,

$$\mathbf{u}_g \cdot \mathbf{n} = \phi\hat{\mathbf{u}} \cdot \mathbf{n}.$$

In the Lagrangian setting, this flux is required to impose a no flux condition at the moving boundary with both the non-relative form (default) and relative form of the advection term. When using the relative form, typically a do-nothing condition would be sufficient to impose no flux; however, to simplify general ALE mesh motion for porous flow equations (i.e., mass balance and porous enthalpy) we automatically provide the mesh motion flux on boundaries for blocks that have mesh motion. Consequently, to impose a zero flux, this term is needed for cancellation.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–

Nd_Table

Syntax

Bc Flux For Mass_Balance ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Open

Syntax

Bc Flux For Mass_Balance ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Open ([Toggle = *toggle*] | [Pressure = *pressure*] | [Entrained_Value = *entrained_value*])

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
pressure	real	0
entrained_value	real	0

Open_Flow

Syntax

Bc Flux For Mass_Balance ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Open_Flow [Toggle = *toggle*]

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–

Polynomial

Syntax

Bc Flux For Mass_Balance ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
variable	“string”	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Flux For Mass_Balance ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
f	“string”	–

User_Field

Syntax

Bc Flux For Mass_Balance ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc Flux For Mass_Balance ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc Flux For Mass_Balance ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

Well_Outflow

Syntax

Bc Flux For Mass_Balance ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Well_Outflow ([Toggle = *toggle*] | Pi = *pi* | Pref = *pref*)

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
pi	"string"	–
pref	"string"	–

Mesh

Constant

Syntax

Bc Flux For Mesh ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | [Valuex = *valuex*] | [Valuey = *valuey*] | [Valuez = *valuez*])

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
valuex	real	0
valuey	real	0
valuez	real	0

Constant_Traction

Syntax

Bc Flux For Mesh ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant_Traction ([Toggle = *toggle*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Constant traction mesh flux

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
x	real	0
y	real	0
z	real	0

Damping

Syntax

Bc Flux For Mesh ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Damping ([Toggle = *toggle*] | [Scale_Factor = *scale_factor*])

Summary

Dampen surface displacements.

Description

$$F = \frac{-2\mu s}{h}\frac{\partial u}{\partial t}$$

where 2μ is the Lame constant, s the optional "scale_factor" parameter, h the mesh size, and u the mesh displacements. This BC is primarily intended for use in conjunction with a rotated BC that enforces the normal mesh displacement, and serves to dampen tangential mesh displacements on the surface.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
scale_factor	real	1

Electric_Traction

Syntax

Bc Flux For Mesh ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Electric_Traction ([Toggle = *toggle*] | [Sign = *sign*])

Summary

Electric traction mesh flux

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
sign	real	1

Penalized_Normal_Displacement

Syntax

Bc Flux For Mesh ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Penalized_Normal_Displacement ([Toggle = *toggle*] | [Stiffness_Factor = *stiffness_factor*])

Summary

Penalized normal flux

Description

Enforces a weak zero-normal-displacement condition by applying a mesh flux of

$$\vec{F} = -2\mu S_f \left(\vec{D} \cdot \vec{n}\right) \vec{n}$$

Where 2μ is the mesh `TWO_MU` value, and S_f is the stiffness factor.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
stiffness_factor	real	1e+06

Pressure

Syntax

Bc Flux For Mesh ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Pressure ([Toggle = *toggle*] | [User_Field_Name = *user_field_name*] | [Use_Bulk_Node = *use_bulk_node*] | [P = *p*] | [C_T = *c_t*] | [C_X = *c_x*] | [C_Y = *c_y*] | [C_Z = *c_z*] | [C_Cos_W = *c_cos_w*] | [C_Sin_W = *c_sin_w*] | [C1_Cos = *c1_cos*] | [C1_Sin = *c1_sin*])

Summary

Pressure mesh flux

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
user_field_name	"string"	–
use_bulk_node	"string"	–
p	real	0
c_t	real	0
c_x	real	0
c_y	real	0
c_z	real	0
c_cos_w	real	0
c_sin_w	real	0
c1_cos	real	0
c1_sin	real	0

Pressure_User_Function

Syntax

Bc Flux For Mesh ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Pressure_User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Pressure user function mesh flux

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1

Shear_Free

Syntax

Bc Flux For Mesh ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Shear_Free [Toggle = *toggle*]

Summary

Shear-free traction mesh flux

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–

Specified_Normal_Velocity

Syntax

Bc Flux For Mesh ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Specified_Normal_Velocity ([Toggle = *toggle*] | [Stiffness_Factor = *stiffness_factor*])

Summary

Penalized normal mesh velocity

Description

Enforces a weak specified normal velocity condition by applying a mesh flux of

$$\vec{F} = 2\mu S_f \left(V_m - \dot{\vec{D}} \cdot \vec{n} \right) \vec{n}$$

Where 2μ is the mesh `TWO_MU` value, and S_f is the stiffness factor. The normal mesh velocity, V_m is provided by the `NORMAL_MESH_VELOCITY` expression on the surface (scalar, positive for outward mesh velocity).

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
stiffness_factor	real	1e+06

Transient_Traction

Syntax

Bc Flux For Mesh ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Transient_Traction ([Toggle = *toggle*] | [A_X = *a_x*] | [A_Y = *a_y*] | [A_Z = *a_z*] | [B_X = *b_x*] | [B_Y = *b_y*] | [B_Z = *b_z*])

Summary

Transient traction mesh flux

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
a_x	real	0
a_y	real	0
a_z	real	0
b_x	real	0
b_y	real	0
b_z	real	0

Vector_String_Function

Syntax

Bc Flux For Mesh ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Vector_String_Function ([Toggle = *toggle*] | F_X = *f_x* | F_Y = *f_y* | [F_Z = *f_z*])

Summary

Value from a string function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f_x	"string"	–
f_y	"string"	–
f_z	"string"	–

Mesh_X

Constant

Syntax

Bc Flux For Mesh_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | Value = *value*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
value	real	–

Exponential

Syntax

Bc Flux For Mesh_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Flux For Mesh_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Bc Flux For Mesh_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc Flux For Mesh_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Flux For Mesh_X ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f	"string"	–

User_Field

Syntax

Bc Flux For Mesh_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc Flux For Mesh_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc Flux For Mesh_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

Mesh_Y

Constant

Syntax

Bc Flux For Mesh_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | Value = *value*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
value	real	–

Exponential

Syntax

Bc Flux For Mesh_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Flux For Mesh_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
global_name	“string”	–
component	integer	0

Nd_Table

Syntax

Bc Flux For Mesh_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc Flux For Mesh_Y ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Flux For Mesh_Y ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f	"string"	–

User_Field

Syntax

Bc Flux For Mesh_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc Flux For Mesh_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc Flux For Mesh_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

Mesh_Z

Constant

Syntax

Bc Flux For Mesh_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | Value = *value*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
value	real	–

Exponential

Syntax

Bc Flux For Mesh_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Flux For Mesh_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
global_name	“string”	–
component	integer	0

Nd_Table

Syntax

Bc Flux For Mesh_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc Flux For Mesh_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Flux For Mesh_Z ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f	"string"	–

User_Field

Syntax

Bc Flux For Mesh_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc Flux For Mesh_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc Flux For Mesh_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

Mixture_Fraction

Constant

Syntax

Bc Flux For Mixture_Fraction ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | *{flux | value}* = *flux*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
flux	real	–

Exponential

Syntax

Bc Flux For Mixture_Fraction ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Flux For Mixture_Fraction ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
global_name	“string”	–
component	integer	0

Nd_Table

Syntax

Bc Flux For Mixture_Fraction ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Open_Flow_Convection

Syntax

Bc Flux For Mixture_Fraction ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Open_Flow_Convection ([Toggle = *toggle*] | Reference_Value = *Reference_Value*)

Summary

Open flow convection BC for the mixture fraction equation

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Reference_Value	real	–

Polynomial

Syntax

Bc Flux For Mixture_Fraction ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Flux For Mixture_Fraction ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f	"string"	–

User_Field

Syntax

Bc Flux For Mixture_Fraction ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc Flux For Mixture_Fraction ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc Flux For Mixture_Fraction ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
Name	“string”	–
plugin_parameters	“string”	–

Momentum

Capillary

Syntax

Bc Flux For Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Capillary [Toggle = *toggle*]

Summary

LS capillary momentum flux

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–

Capillary_Gap

Syntax

Bc Flux For Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Capillary_Gap ([Toggle = *toggle*] | V_W = *v_w* | G = *g* | Theta = *theta*)

Summary

This model has been deprecated

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
v_w	real	–
g	real	–
theta	real	–

Constant

Syntax

Bc Flux For Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
x	real	0
y	real	0
z	real	0

Constant_Traction

Syntax

Bc Flux For Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant_Traction ([Toggle = *toggle*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
x	real	0
y	real	0
z	real	0

Darcy_Slip

Syntax

Bc Flux For Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Darcy_Slip ([Toggle = *toggle*] | [Vs_X = *vs_x*] | [Vs_Y = *vs_y*] | [Vs_Z = *vs_z*] | K = *k*)

Summary

Darcy slip momentum flux

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
vs_x	real	0
vs_y	real	0
vs_z	real	0
k	real	–

Electric_Traction

Syntax

Bc Flux For Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Electric_Traction ([Toggle = *toggle*] | [Sign = *sign*])

Summary

Electric traction momentum flux

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
sign	real	1

Equilibrium_Capillary_Gap

Syntax

Bc Flux For Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Equilibrium_Capillary_Gap ([Toggle = *toggle*] | Theta = *theta*)

Summary

This model has been deprecated

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
theta	real	–

Exponential_Vapor_Recoil_Pressure

Syntax

Bc Flux For Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential_Vapor_Recoil_Pressure ([Toggle = *toggle*] | Tboil = *tboil* | L = *l* | Pa = *pa* | [Beta = *beta*] | Mw = *mw* | [Tl = *tl*] | [Th = *th*])

Summary

Exponential vapor recoil pressure momentum flux

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
tboil	real	–
l	real	–
pa	real	–
beta	real	1
mw	real	–
tl	real	-1
th	real	-1

Fixed_Curvature

Syntax

Bc Flux For Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Fixed_Curvature ([Toggle = *toggle*] | Curvature = *curvature*)

Summary

This model has been deprecated and removed from the code.

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
curvature	real	–

Flow_Hydrostatic

Syntax

Bc Flux For Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Flow_Hydrostatic ([Toggle = *toggle*] | [P_Ref = *p_ref*] | [Gx = *gx*] | [Gy = *gy*] | [Gz = *gz*])

Summary

Flow hydrostatic momentum flux

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
p_ref	real	0
gx	real	0
gy	real	0
gz	real	0

Free_Open_Flow

Syntax

Bc Flux For Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Free_Open_Flow ([Toggle = *toggle*] | Pressure = *pressure*)

Summary

Open flow for momentum; apply the full stress

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
pressure	real	–

Lens_Particles_Pressure

Syntax

Bc Flux For Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Lens_Particles_Pressure ([Toggle = *toggle*] | Src_X = *src_x* | Dir_X = *dir_x* | Vel_X = *vel_x* | Src_Y = *src_y* | Dir_Y = *dir_y* | Vel_Y = *vel_y* | [Src_Z = *src_z*] | [Dir_Z = *dir_z*] | [Vel_Z = *vel_z*] | Massflowrate = *massFlowRate* | T_Liq = *T_liq* | T_Solid = *T_solid* | [T_On = *t_on*] | [T_Off = *t_off*] | Particlespeed = *particleSpeed* | Radius = *radius*)

Summary

LENS particles pressure momentum flux

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
src_x	real	–
dir_x	real	–
vel_x	real	–
src_y	real	–
dir_y	real	–
vel_y	real	–
src_z	real	0
dir_z	real	0
vel_z	real	0
massFlowRate	real	–
T_liq	real	–
T_solid	real	–
t_on	real	0
t_off	real	1.798e+308
particleSpeed	real	–
radius	real	–

Ls_Capillary

Syntax

Bc Flux For Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Ls_Capillary [Toggle = *toggle*]

Summary

LS capillary momentum flux

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–

Ls_Oriented_Slip

Syntax

Bc Flux For Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Ls_Oriented_Slip ([Toggle = *toggle*] | [Vs_X = *vs_x*] | [Vs_Y = *vs_y*] | [Vs_Z = *vs_z*] | [Beta_Normal_A = *beta_normal_a*] | [Beta_Tangent_A = *beta_tangent_a*] | [Beta_Normal_B = *beta_normal_b*] | [Beta_Tangent_B = *beta_tangent_b*])

Summary

Oriented slip momentum flux

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
vs_x	real	0
vs_y	real	0
vs_z	real	0
beta_normal_a	real	0
beta_tangent_a	real	0
beta_normal_b	real	0
beta_tangent_b	real	0

Node_Normal_Capillary

Syntax

Bc Flux For Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Node_Normal_Capillary [Toggle = *toggle*]

Summary

This model has been deprecated and removed from the code.

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–

Non_Ibp_Pressure

Syntax

Bc Flux For Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Non_Ibp_Pressure ([Toggle = *toggle*] | [Pressure = *pressure*])

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
pressure	"string"	–

Open_Flow

Syntax

Bc Flux For Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Open_Flow ([Toggle = *toggle*] | [Pressure = *pressure*] | [Total_Pressure = *total_pressure*])

Summary

Open flow for momentum; enforce normal gradient to be zero

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
pressure	real	0
total_pressure	real	0

Oriented_Slip

Syntax

Bc Flux For Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Oriented_Slip ([Toggle = *toggle*] | [Vs_X = *vs_x*] | [Vs_Y = *vs_y*] | [Vs_Z = *vs_z*] | [Vs_Expr = *vs_expr*] | Beta_Normal = *beta_normal* | Beta_Tangent = *beta_tangent*)

Summary

Oriented slip momentum flux. Essentially a sum of the Navier-Slip condition and a penalty condition for enforcing velocity in the normal direction.

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
vs_x	real	0
vs_y	real	0
vs_z	real	0
vs_expr	"string"	–
beta_normal	real	–
beta_tangent	real	–

Pressure

Syntax

Bc Flux For Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Pressure ([Toggle = *toggle*] | [User_Field_Name = *user_field_name*] | [Use_Bulk_Node = *use_bulk_node*] | [P = *p*] | [C_T = *c_t*] | [C_X = *c_x*] | [C_Y = *c_y*] | [C_Z = *c_z*] | [C_Cos_W = *c_cos_w*] | [C_Sin_W = *c_sin_w*] | [C1_Cos = *c1_cos*] | [C1_Sin = *c1_sin*])

Summary

Pressure momentum flux

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is

active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
user_field_name	"string"	–
use_bulk_node	"string"	–
p	real	0
c_t	real	0
c_x	real	0
c_y	real	0
c_z	real	0
c_cos_w	real	0
c_sin_w	real	0
c1_cos	real	0
c1_sin	real	0

Pressure_User_Function

Syntax

Bc Flux For Momentum ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Pressure_User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Pressure user function momentum flux

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1

Projected_Capillary

Syntax

Bc Flux For Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Projected_Capillary [Toggle = *toggle*]

Summary

Projected capillary momentum flux

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–

Reconstructed_Curvature

Syntax

Bc Flux For Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Reconstructed_Curvature ([Toggle = *toggle*] | [Ls_Size = *ls_size*] | [Fixed_Curvature = *fixed_curvature*])

Summary

This model has been deprecated and removed from the code.

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
ls_size	integer	0
fixed_curvature	real	-1

Simple_Spring_Force

Syntax

Bc Flux For Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Simple_Spring_Force ([Toggle = *toggle*] | [Kx = *kx*] | [Ky = *ky*] | [Kz = *kz*] | [X0 = *x0*] | [Y0 = *y0*] | [Z0 = *z0*])

Summary

This model has been deprecated and removed from the code.

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
kx	real	0
ky	real	0
kz	real	0
x0	real	0
y0	real	0
z0	real	0

Slip

Syntax

Bc Flux For Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Slip ([Toggle = *toggle*] | [Vs_X = *vs_x*] | [Vs_Y = *vs_y*] | [Vs_Z = *vs_z*] | Beta = *beta*)

Summary

Slip momentum flux

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
vs_x	real	0
vs_y	real	0
vs_z	real	0
beta	real	–

Slip_Length

Syntax

Bc Flux For Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Slip_Length ([Toggle = *toggle*] | [Vs_X = *vs_x*] | [Vs_Y = *vs_y*] | [Vs_Z = *vs_z*] | Beta = *beta*)

Summary

Slip length momentum flux

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
vs_x	real	0
vs_y	real	0
vs_z	real	0
beta	real	–

Spring_Force

Syntax

Bc Flux For Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Spring_Force ([Toggle = *toggle*] | Mult = *Mult*)

Summary

Spring force for momentum boundary condition

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Mult	real	–

Symmetry_Flow

Syntax

Bc Flux For Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Symmetry_Flow [Toggle = *toggle*]

Summary

Symmetry flow momentum flux

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–

Transient_Traction

Syntax

Bc Flux For Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Transient_Traction ([Toggle = *toggle*] | [A_X = *a_x*] | [A_Y = *a_y*] | [A_Z = *a_z*] | [B_X = *b_x*] | [B_Y = *b_y*] | [B_Z = *b_z*])

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
a_x	real	0
a_y	real	0
a_z	real	0
b_x	real	0
b_y	real	0
b_z	real	0

Vapor_Recoil_Pressure

Syntax

Bc Flux For Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Vapor_Recoil_Pressure ([Toggle = *toggle*] | Tboil = *tboil*)

Summary

Vapor recoil pressure momentum flux

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
tboil	real	–

Vector_String_Function

Syntax

Bc Flux For Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Vector_String_Function ([Toggle = *toggle*] | F_X = *f_x* | F_Y = *f_y* | [F_Z = *f_z*])

Summary

Value from a string function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f_x	"string"	–
f_y	"string"	–
f_z	"string"	–

Momentum_X

Constant

Syntax

Bc Flux For Momentum_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | Value = *value*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
value	real	–

Exponential

Syntax

Bc Flux For Momentum_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Flux For Momentum_X ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Bc Flux For Momentum_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc Flux For Momentum_X ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Flux For Momentum_X ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f	"string"	–

User_Field

Syntax

Bc Flux For Momentum_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc Flux For Momentum_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc Flux For Momentum_X ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

Momentum_Y

Constant

Syntax

Bc Flux For Momentum_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | Value = *value*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
value	real	–

Exponential

Syntax

Bc Flux For Momentum_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Flux For Momentum_Y ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Bc Flux For Momentum_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc Flux For Momentum_Y ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Flux For Momentum_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f	"string"	–

User_Field

Syntax

Bc Flux For Momentum_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc Flux For Momentum_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc Flux For Momentum_Y ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

Momentum_Z

Constant

Syntax

Bc Flux For Momentum_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | Value = *value*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
value	real	–

Exponential

Syntax

Bc Flux For Momentum_Z ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Flux For Momentum_Z ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Bc Flux For Momentum_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc Flux For Momentum_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Flux For Momentum_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f	"string"	–

User_Field

Syntax

Bc Flux For Momentum_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
scaling	real	1
global_var	“string”	–
time_function	“string”	–

User_Function

Syntax

Bc Flux For Momentum_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc Flux For Momentum_Z ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

Porous_Enthalpy

Constant

Syntax

Bc Flux For Porous_Enthalpy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | Value = *value*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
value	real	–

Convective_Outflow

Syntax

Bc Flux For Porous_Enthalpy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Convective_Outflow ([Toggle = *toggle*] | [Ref_H = *ref_h*])

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
ref_h	real	0

Enclosure_Radiation

Syntax

Bc Flux For Porous_Enthalpy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Enclosure_Radiation ([Toggle = *toggle*] | [Multiplier = *multiplier*] | Enclosure = *enclosure*)

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
multiplier	real	1
enclosure	“string”	–

Exponential

Syntax

Bc Flux For Porous_Enthalpy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the

constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Fluid_Robin_Coupled

Syntax

Bc Flux For Porous_Enthalpy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Fluid_Robin_Coupled ([Toggle = *toggle*] | Enthalpy_Field = *enthalpy_field* | Enthalpy_Diffusive_Flux_Field = *enthalpy_diffusive_flux_field* | Coeff_Field = *coeff_field*)

Summary

Boundary condition for loose coupling of enthalpy to a free fluid region

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
enthalpy_field	"string"	–
enthalpy_diffusive_flux_field	"string"	–
coeff_field	"string"	–

Fluid_Robin_Coupled_One_Region

Syntax

Bc Flux For Porous_Enthalpy ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Fluid_Robin_Coupled_One_Region [Toggle = *toggle*]

Summary

Boundary condition for loose coupling of enthalpy to a free fluid region

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–

Fluid_Robin_Coupled_With_Solid_Convection

Syntax

Bc Flux For Porous_Enthalpy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Fluid_Robin_Coupled_With_Solid_Convection ([Toggle = *toggle*] | Enthalpy_Field = *enthalpy_field* | Enthalpy_Diffusive_Flux_Field = *enthalpy_diffusive_flux_field* | Coeff_Field = *coeff_field*)

Summary

Boundary condition for loose coupling of enthalpy to a free fluid region

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
enthalpy_field	“string”	–
enthalpy_diffusive_flux_field	“string”	–
coeff_field	“string”	–

Fluid_Robin_Coupled_With_Solid_Convection_One_Region

Syntax

Bc Flux For Porous_Enthalpy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Fluid_Robin_Coupled_With_Solid_Convection_One_Region [Toggle = *toggle*]

Summary

Boundary condition for loose coupling of enthalpy to a free fluid region

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–

Fluid_Solid_Convection_Coupled

Syntax

Bc Flux For Porous_Enthalpy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Fluid_Solid_Convection_Coupled ([Toggle = *toggle*] | [Multiplier = *multiplier*] | Fluid_Temperature_Field = *fluid_temperature_field*)

Summary

Boundary condition for loose coupling of enthalpy to a free fluid region

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
multiplier	real	1
fluid_temperature_field	"string"	–

Fluid_Solid_Convection_Coupled_One_Region

Syntax

Bc Flux For Porous_Enthalpy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Fluid_Solid_Convection_Coupled_One_Region ([Toggle = *toggle*] | [Multiplier = *multiplier*])

Summary

Boundary condition for loose coupling of enthalpy to a free fluid region

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
multiplier	real	1

Generalized_Nat_Conv

Syntax

Bc Flux For Porous_Enthalpy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Generalized_Nat_Conv ([Toggle = *toggle*] | [Multiplier = *multiplier*])

Summary

Generalized natural convection flux BC

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
multiplier	real	1

Generalized_Rad

Syntax

Bc Flux For Porous_Enthalpy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Generalized_Rad ([Toggle = *toggle*] | [Multiplier = *multiplier*] | [T_Ref = *t_ref*])

Summary

Radiative energy flux, with values for emissivity, form factor and irradiation taken from expressions

Description

Flux is:

$$M\epsilon f\left(\sigma T^4 - I\right)$$

M is MULTIPLIER. σ is the Stefan-Boltzmann constant. ϵ and F are expressions for emissivity and form factor. T is surface temperature. I is the expression for irradiation. If I does not already exist, it will be constructed as σT_{ref}^4, where T_{ref} is either provided by the BC Rad Reference Temperature expression or the inline T_{ref} parameter (constant or string function).

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
multiplier	real	1
t_ref	“string”	–

Global

Syntax

Bc Flux For Porous_Enthalpy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
global_name	“string”	–
component	integer	0

Material_Phase_Sum_Open

Syntax

Bc Flux For Porous_Enthalpy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Material_Phase_Sum_Open ([Toggle = *toggle*] | [Pressure = *pressure*] | [Entrained_Value = *entrained_value*])

Summary

Open boundary condition for a single temperature model, wherein the phases are assumed to be in thermal equilibrium. The advective flux of enthalpy for

each phase is summed at the open.Must provide the BOUNDARY_ENTRAINED_ENTHALPY expression for entrainment.

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
pressure	real	0
entrained_value	real	0

Mesh_Motion

Syntax

Bc Flux For Porous_Enthalpy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Mesh_Motion [Toggle = *toggle*]

Summary

Moving boundary condition for Arbitrary Lagrangian-Eulerian (ALE) mesh motion in relative flux form. Accounts for the boundary advective flux due to general mesh motion.

Description

Applies the following boundary flux

$$-\int q\hat{\mathbf{u}}\cdot\mathbf{n}dS,$$

where $\hat{u}$ is the mesh velocity and where the integrated scalar q depends on the equation. For the gas phase `porous_enthalpy` equation with a `transported`

`enthalpy = porous`, $q = \phi\rho_g h_g$. Similarly, for the gas phase `mass_balance` equation, $q = \phi\rho_g$ is the bulk mass density.

This flux boundary condition is automatically applied to the mass balance and porous enthalpy equations when the `use relative mass flux` porous flow option is turned on.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle
If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–

Moving_Boundary

Syntax

Bc Flux For Porous_Enthalpy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Moving_Boundary [Toggle = *toggle*]

Summary

No penetration moving boundary condition, valid for Lagrangian mesh motion in non-relative flux form or relative flux form.

Description

Applies the following boundary flux

$$\int q\hat{\mathbf{u}} \cdot \mathbf{n} dS,$$

where $\hat{u}$ is the mesh velocity and where the integrated scalar q depends on the equation. If we consider the gas phase `porous_enthalpy` equation with a `transported enthalpy = porous`, then $q = \phi\rho_g h_g$. Similarly, for the gas phase `mass_balance` equation, $q = \phi\rho_g$ is the bulk mass density.

If we consider the gas phase `mass_balance` equation, with a gas phase porous velocity relative to the moving solid phase, then physically this boundary condition is derived from applying a no flux condition at moving boundary i.e.,

$$\mathbf{u}_g \cdot \mathbf{n} = \phi \hat{\mathbf{u}} \cdot \mathbf{n}.$$

In the Lagrangian setting, this flux is required to impose a no flux condition at the moving boundary with both the non-relative form (default) and relative form of the advection term. When using the relative form, typically a do-nothing condition would be sufficient to impose no flux; however, to simplify general ALE mesh motion for porous flow equations (i.e., mass balance and porous enthalpy) we automatically provide the mesh motion flux on boundaries for blocks that have mesh motion. Consequently, to impose a zero flux, this term is needed for cancellation.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle
If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–

Nd_Table

Syntax

Bc Flux For Porous_Enthalpy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

One_Temp_Multiphase_Open

Syntax

Bc Flux For Porous_Enthalpy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = One_Temp_Multiphase_Open ([Toggle = *toggle*] | [Pressure = *pressure*] | [Entrained_Value = *entrained_value*])

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
pressure	real	0
entrained_value	real	0

Open

Syntax

Bc Flux For Porous_Enthalpy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Open ([Toggle = *toggle*] | [Pressure = *pressure*] | [Entrained_Value = *entrained_value*])

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
pressure	real	0
entrained_value	real	0

Polynomial

Syntax

Bc Flux For Porous_Enthalpy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
variable	“string”	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Flux For Porous_Enthalpy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
f	“string”	–

User_Field

Syntax

Bc Flux For Porous_Enthalpy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
scaling	real	1
global_var	“string”	–
time_function	“string”	–

User_Function

Syntax

Bc Flux For Porous_Enthalpy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc Flux For Porous_Enthalpy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

User_Vector_Field

Syntax

Bc Flux For Porous_Enthalpy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Vector_Field ([Toggle = *toggle*] | Name = *name* | [Multiplier = *multiplier*])

Summary

User vector field enthalpy flux

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
multiplier	real	1

Porous_Species

Constant

Syntax

Bc Flux For Porous_Species ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | *{flux | value}* = *flux*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
flux	real	–

Exponential

Syntax

Bc Flux For Porous_Species ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Flux For Porous_Species ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
global_name	"string"	–
component	integer	0

Moving_Boundary

Syntax

Bc Flux For Porous_Species ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Moving_Boundary [Toggle = *toggle*]

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–

Nd_Table

Syntax

Bc Flux For Porous_Species ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Open_Flow

Syntax

Bc Flux For Porous_Species ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Open_Flow [Toggle = *toggle*]

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–

Polynomial

Syntax

Bc Flux For Porous_Species ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Flux For Porous_Species ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name.

Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
f	“string”	–

User_Field

Syntax

Bc Flux For Porous_Species ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc Flux For Porous_Species ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc Flux For Porous_Species ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

Solid

Constant

Syntax

Bc Flux For Solid ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | [Valuex = *valuex*] | [Valuey = *valuey*] | [Valuez = *valuez*])

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
valuex	real	0
valuey	real	0
valuez	real	0

Constant_Traction

Syntax

Bc Flux For Solid ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant_Traction ([Toggle = *toggle*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Constant traction solid flux

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
x	real	0
y	real	0
z	real	0

Electric_Traction

Syntax

Bc Flux For Solid ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Electric_Traction ([Toggle = *toggle*] | [Sign = *sign*])

Summary

Electric traction solid flux

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
sign	real	1

Pressure

Syntax

Bc Flux For Solid ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Pressure ([Toggle = *toggle*] | [User_Field_Name = *user_field_name*] | [Use_Bulk_Node = *use_bulk_node*] | [P = *p*] | [C_T = *c_t*] | [C_X = *c_x*] | [C_Y = *c_y*] | [C_Z = *c_z*] | [C_Cos_W = *c_cos_w*] | [C_Sin_W = *c_sin_w*] | [C1_Cos = *c1_cos*] | [C1_Sin = *c1_sin*])

Summary

Pressure solid flux

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
user_field_name	"string"	–
use_bulk_node	"string"	–
p	real	0
c_t	real	0
c_x	real	0
c_y	real	0
c_z	real	0
c_cos_w	real	0
c_sin_w	real	0
c1_cos	real	0
c1_sin	real	0

Pressure_User_Function

Syntax

Bc Flux For Solid ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Pressure_User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Pressure user function solid flux

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1

Shear_Free

Syntax

Bc Flux For Solid ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Shear_Free [Toggle = *toggle*]

Summary

Shear-free traction solid flux

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–

Spring_Force

Syntax

Bc Flux For Solid ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Spring_Force ([Toggle = *toggle*] | Mult = *Mult*)

Summary

Spring force solid flux

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Mult	real	–

Transient_Traction

Syntax

Bc Flux For Solid ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Transient_Traction ([Toggle = *toggle*] | [A_X = *a_x*] | [A_Y = *a_y*] | [A_Z = *a_z*] | [B_X = *b_x*] | [B_Y = *b_y*] | [B_Z = *b_z*])

Summary

Transient traction solid flux

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
a_x	real	0
a_y	real	0
a_z	real	0
b_x	real	0
b_y	real	0
b_z	real	0

Vector_String_Function

Syntax

Bc Flux For Solid ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Vector_String_Function ([Toggle = *toggle*] | F_X = *f_x* | F_Y = *f_y* | [F_Z = *f_z*])

Summary

Value from a string function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
f_x	“string”	–
f_y	“string”	–
f_z	“string”	–

Solid_X

Constant

Syntax

Bc Flux For Solid_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | Value = *value*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
value	real	–

Exponential

Syntax

Bc Flux For Solid_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Flux For Solid_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
global_name	“string”	–
component	integer	0

Nd_Table

Syntax

Bc Flux For Solid_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc Flux For Solid_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Flux For Solid_X ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f	"string"	–

User_Field

Syntax

Bc Flux For Solid_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc Flux For Solid_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc Flux For Solid_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

Solid_Y

Constant

Syntax

Bc Flux For Solid_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | Value = *value*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
value	real	–

Exponential

Syntax

Bc Flux For Solid_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Flux For Solid_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
global_name	“string”	–
component	integer	0

Nd_Table

Syntax

Bc Flux For Solid_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc Flux For Solid_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Flux For Solid_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f	"string"	–

User_Field

Syntax

Bc Flux For Solid_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc Flux For Solid_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc Flux For Solid_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

Solid_Z

Constant

Syntax

Bc Flux For Solid_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | Value = *value*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
value	real	–

Exponential

Syntax

Bc Flux For Solid_Z ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Flux For Solid_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Bc Flux For Solid_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc Flux For Solid_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Flux For Solid_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f	"string"	–

User_Field

Syntax

Bc Flux For Solid_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc Flux For Solid_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc Flux For Solid_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
Name	“string”	–
plugin_parameters	“string”	–

Sp

Calore_User_Sub

Syntax

Bc Flux For Sp ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Calore_User_Sub (Name = *name* | Type = *type* | [Multiplier = *multiplier*] | [Material_Data_Block = *material_data_block*] | [Data = *data*] | [Scaling_Field = *scaling_field*])

Summary

Values from a Calore user subroutine

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
name	“string”	–
type	“string”	–
multiplier	real	1
material_data_block	“string”	–
data	“string”	–
scaling_field	“string”	–

Constant

Syntax

Bc Flux For Sp ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant *{flux | value}* = *flux*

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
flux	real	–

Exponential

Syntax

Bc Flux For Sp ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Flux For Sp ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global (Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
global_name	"string"	–
component	integer	0

Mixed

Syntax

Bc Flux For Sp ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Mixed ()

Summary

Applies the MIXED scalar flux from the spherical harmonic equations.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–

Polynomial

Syntax

Bc Flux For Sp ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Flux For Sp ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function F = *f*

Summary

Value from a string function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
f	"string"	–

User_Field

Syntax

Bc Flux For Sp ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
name	“string”	–
scaling	real	1
global_var	“string”	–
time_function	“string”	–

User_Function

Syntax

Bc Flux For Sp ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

Species

Constant

Syntax

Bc Flux For Species ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | *{flux | value}* = *flux*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
flux	real	–

Exponential

Syntax

Bc Flux For Species ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Free_Open_Flow

Syntax

Bc Flux For Species ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Free_Open_Flow [Toggle = *toggle*]

Summary

This Flux BC sets the surface normal species flux equal to the diffusive flux.

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–

Global

Syntax

Bc Flux For Species ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
global_name	“string”	–
component	integer	0

Interface

Syntax

Bc Flux For Species ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Interface [Toggle = *toggle*]

Summary

Flux BC model of species for moving interfaces

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–

Nat_Conv

Syntax

Bc Flux For Species ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nat_Conv ([Toggle = *toggle*] | K = *k* | Yinf = *yinf*)

Summary

natconv species flux

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
k	real	–
yinf	real	–

Nd_Table

Syntax

Bc Flux For Species ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Open_Adv_Flow

Syntax

Bc Flux For Species ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Open_Adv_Flow ([Toggle = *toggle*] | [Far_Field_Entrainment_Value = *far_field_entrainment_value*])

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
far_field_entrainment_value	real	0

Polynomial

Syntax

Bc Flux For Species ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Flux For Species ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f	"string"	–

User_Field

Syntax

Bc Flux For Species ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc Flux For Species ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc Flux For Species ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

Voltage

Constant

Syntax

Bc Flux For Voltage ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | *{flux | value}* = *flux*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
flux	real	–

Exponential

Syntax

Bc Flux For Voltage ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Flux For Voltage ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Bc Flux For Voltage ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc Flux For Voltage ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Flux For Voltage ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f	"string"	–

User_Field

Syntax

Bc Flux For Voltage ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc Flux For Voltage ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc Flux For Voltage ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

Bc Fluxbp

Continuity

Constant

Syntax

Bc Fluxbp For Continuity ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant Value = *value*

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
value	real	–

Exponential

Syntax

Bc Fluxbp For Continuity ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
variable	“string”	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Fluxbp For Continuity ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global (Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Bc Fluxbp For Continuity ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc Fluxbp For Continuity ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Fluxbp For Continuity ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function F = *f*

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
f	"string"	–

User_Field

Syntax

Bc Fluxbp For Continuity ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc Fluxbp For Continuity ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Bc Fluxbp For Continuity ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin (Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
Name	“string”	–
plugin_parameters	“string”	–

Cvfem_Momentum

Capillary

Syntax

Bc Fluxbp For Cvfem_Momentum ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Capillary ()

Summary

Momentum capillary flux BP BC

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–

Capillary_Stabilization

Syntax

Bc Fluxbp For Cvfem_Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Capillary_Stabilization ([Mu0 = *mu0*] | [Multiplier = *multiplier*] | [Use_Velocity_Correction = *use_velocity_correction*])

Summary

Momentum capillary stabilization flux BP BC

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
mu0	real	0
multiplier	real	1
use_velocity_correction	integer	–

Constant

Syntax

Bc Fluxbp For Cvfem_Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Valuex = *valuex*] | [Valuey = *valuey*] | [Valuez = *valuez*])

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
valuex	real	0
valuey	real	0
valuez	real	0

Ls_Capillary

Syntax

Bc Fluxbp For Cvfem_Momentum ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Ls_Capillary ()

Summary

Momentum capillary flux BP BC using level set normal

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–

Ls_Capillary_Stabilization

Syntax

Bc Fluxbp For Cvfem_Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Ls_Capillary_Stabilization ([Mu0 = *mu0*] | [Multiplier = *multiplier*] | [Use_Velocity_Correction = *use_velocity_correction*])

Summary

Momentum capillary stabilization flux BP BC using level set normal

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
mu0	real	0
multiplier	real	1
use_velocity_correction	integer	–

Normal_Capillary_Stabilization

Syntax

Bc Fluxbp For Cvfem_Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Normal_Capillary_Stabilization ([Mu0 = *mu0*] | [Multiplier = *multiplier*] | [Use_Velocity_Correction = *use_velocity_correction*])

Summary

Momentum capillary stabilization flux BP BC

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
mu0	real	0
multiplier	real	1
use_velocity_correction	integer	–

Vector_String_Function

Syntax

Bc Fluxbp For Cvfem_Momentum ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Vector_String_Function (F_X = *f_x* | F_Y = *f_y* | [F_Z = *f_z*])

Summary

Value from a string function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
f_x	“string”	–
f_y	“string”	–
f_z	“string”	–

Cvfem_Momentum_X

Constant

Syntax

Bc Fluxbp For Cvfem_Momentum_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant Value = *value*

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
value	real	–

Exponential

Syntax

Bc Fluxbp For Cvfem_Momentum_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Fluxbp For Cvfem_Momentum_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global (Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Bc Fluxbp For Cvfem_Momentum_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc Fluxbp For Cvfem_Momentum_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Fluxbp For Cvfem_Momentum_X ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function F = *f*

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about

valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
f	"string"	–

User_Field

Syntax

Bc Fluxbp For Cvfem_Momentum_X ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc Fluxbp For Cvfem_Momentum_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Bc Fluxbp For Cvfem_Momentum_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin (Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
Name	"string"	–
plugin_parameters	"string"	–

Cvfem_Momentum_Y

Constant

Syntax

Bc Fluxbp For Cvfem_Momentum_Y ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant Value = *value*

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
value	real	–

Exponential

Syntax

Bc Fluxbp For Cvfem_Momentum_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
variable	“string”	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Fluxbp For Cvfem_Momentum_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global (Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Bc Fluxbp For Cvfem_Momentum_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc Fluxbp For Cvfem_Momentum_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Fluxbp For Cvfem_Momentum_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function F = *f*

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
f	"string"	–

User_Field

Syntax

Bc Fluxbp For Cvfem_Momentum_Y ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc Fluxbp For Cvfem_Momentum_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Bc Fluxbp For Cvfem_Momentum_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin (Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
Name	“string”	–
plugin_parameters	“string”	–

Cvfem_Momentum_Z

Constant

Syntax

Bc Fluxbp For Cvfem_Momentum_Z ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant Value = *value*

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
value	real	–

Exponential

Syntax

Bc Fluxbp For Cvfem_Momentum_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
variable	“string”	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Fluxbp For Cvfem_Momentum_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global (Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Bc Fluxbp For Cvfem_Momentum_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc Fluxbp For Cvfem_Momentum_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Fluxbp For Cvfem_Momentum_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function F = *f*

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
f	"string"	–

User_Field

Syntax

Bc Fluxbp For Cvfem_Momentum_Z ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc Fluxbp For Cvfem_Momentum_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Bc Fluxbp For Cvfem_Momentum_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin (Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
Name	“string”	–
plugin_parameters	“string”	–

Energy

Surface_Stabilization

Syntax

Bc Fluxbp For Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Surface_Stabilization D = *d*

Summary

surface diffusion term for energy

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
d	real	–

Mesh

Capillary

Syntax

Bc Fluxbp For Mesh ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Capillary ()

Summary

Mesh capillary flux BP BC

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–

Constant

Syntax

Bc Fluxbp For Mesh ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Valuex = *valuex*] | [Valuey = *valuey*] | [Valuez = *valuez*])

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
valuex	real	0
valuey	real	0
valuez	real	0

Vector_String_Function

Syntax

Bc Fluxbp For Mesh ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Vector_String_Function (F_X = *f_x* | F_Y = *f_y* | [F_Z = *f_z*])

Summary

Value from a string function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
f_x	“string”	–
f_y	“string”	–
f_z	“string”	–

Mesh_X

Constant

Syntax

Bc Fluxbp For Mesh_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant Value = *value*

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
value	real	–

Exponential

Syntax

Bc Fluxbp For Mesh_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Fluxbp For Mesh_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global (Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Bc Fluxbp For Mesh_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc Fluxbp For Mesh_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Fluxbp For Mesh_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function F = *f*

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about

valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
f	"string"	–

User_Field

Syntax

Bc Fluxbp For Mesh_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc Fluxbp For Mesh_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Bc Fluxbp For Mesh_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin (Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
Name	“string”	–
plugin_parameters	“string”	–

Mesh_Y

Constant

Syntax

Bc Fluxbp For Mesh_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant Value = *value*

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
value	real	–

Exponential

Syntax

Bc Fluxbp For Mesh_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
variable	“string”	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Fluxbp For Mesh_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global (Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
global_name	“string”	–
component	integer	0

Nd_Table

Syntax

Bc Fluxbp For Mesh_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc Fluxbp For Mesh_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Fluxbp For Mesh_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function F = *f*

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
f	"string"	–

User_Field

Syntax

Bc Fluxbp For Mesh_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc Fluxbp For Mesh_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Bc Fluxbp For Mesh_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin (Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
Name	“string”	–
plugin_parameters	“string”	–

Mesh_Z

Constant

Syntax

Bc Fluxbp For Mesh_Z ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant Value = *value*

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
value	real	–

Exponential

Syntax

Bc Fluxbp For Mesh_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
variable	“string”	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Fluxbp For Mesh_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global (Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Bc Fluxbp For Mesh_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc Fluxbp For Mesh_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Fluxbp For Mesh_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function F = *f*

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
f	"string"	–

User_Field

Syntax

Bc Fluxbp For Mesh_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc Fluxbp For Mesh_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Bc Fluxbp For Mesh_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin (Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
Name	“string”	–
plugin_parameters	“string”	–

Momentum

Capillary

Syntax

Bc Fluxbp For Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Capillary [Use_Node_Normal = *use_node_normal*]

Summary

Momentum capillary flux BP BC

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
use_node_normal	integer	0

Capillary_Stabilization

Syntax

Bc Fluxbp For Momentum ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Capillary_Stabilization ([Mu0 = *mu0*] | [Multiplier = *multiplier*] | [Use_Velocity_Correction = *use_velocity_correction*])

Summary

Momentum capillary stabilization flux BP BC

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
mu0	real	0
multiplier	real	1
use_velocity_correction	integer	–

Constant

Syntax

Bc Fluxbp For Momentum ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Valuex = *valuex*] | [Valuey = *valuey*] | [Valuez = *valuez*])

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
valuex	real	0
valuey	real	0
valuez	real	0

Ls_Capillary

Syntax

Bc Fluxbp For Momentum ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Ls_Capillary ()

Summary

Momentum capillary flux BP BC using level set normal

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–

Ls_Capillary_Stabilization

Syntax

Bc Fluxbp For Momentum ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Ls_Capillary_Stabilization ([Mu0 = *mu0*] | [Multiplier = *multiplier*] | [Use_Velocity_Correction = *use_velocity_correction*])

Summary

Momentum capillary stabilization flux BP BC using level set normal

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
mu0	real	0
multiplier	real	1
use_velocity_correction	integer	–

Normal_Capillary_Stabilization

Syntax

Bc Fluxbp For Momentum ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Normal_Capillary_Stabilization ([Mu0 = *mu0*] | [Multiplier = *multiplier*] | [Use_Velocity_Correction = *use_velocity_correction*])

Summary

Momentum capillary stabilization flux BP BC

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
mu0	real	0
multiplier	real	1
use_velocity_correction	integer	–

Smoothed_Capillary

Syntax

Bc Fluxbp For Momentum ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Smoothed_Capillary ()

Summary

Momentum capillary flux BP BC

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–

Vector_String_Function

Syntax

Bc Fluxbp For Momentum ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Vector_String_Function (F_X = *f_x* | F_Y = *f_y* | [F_Z = *f_z*])

Summary

Value from a string function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
f_x	“string”	–
f_y	“string”	–
f_z	“string”	–

Momentum_X

Constant

Syntax

Bc Fluxbp For Momentum_X ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant Value = *value*

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
value	real	–

Exponential

Syntax

Bc Fluxbp For Momentum_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
variable	“string”	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Fluxbp For Momentum_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global (Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Bc Fluxbp For Momentum_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc Fluxbp For Momentum_X ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Fluxbp For Momentum_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function F = *f*

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
f	"string"	–

User_Field

Syntax

Bc Fluxbp For Momentum_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc Fluxbp For Momentum_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Bc Fluxbp For Momentum_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin (Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
Name	“string”	–
plugin_parameters	“string”	–

Momentum_Y

Constant

Syntax

Bc Fluxbp For Momentum_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant Value = *value*

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
value	real	–

Exponential

Syntax

Bc Fluxbp For Momentum_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
variable	“string”	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Fluxbp For Momentum_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global (Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Bc Fluxbp For Momentum_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc Fluxbp For Momentum_Y ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Fluxbp For Momentum_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function F = *f*

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
f	"string"	–

User_Field

Syntax

Bc Fluxbp For Momentum_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc Fluxbp For Momentum_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Bc Fluxbp For Momentum_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin (Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
Name	“string”	–
plugin_parameters	“string”	–

Momentum_Z

Constant

Syntax

Bc Fluxbp For Momentum_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant Value = *value*

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
value	real	–

Exponential

Syntax

Bc Fluxbp For Momentum_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
variable	“string”	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Fluxbp For Momentum_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global (Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Bc Fluxbp For Momentum_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc Fluxbp For Momentum_Z ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Fluxbp For Momentum_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function F = *f*

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
f	"string"	–

User_Field

Syntax

Bc Fluxbp For Momentum_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc Fluxbp For Momentum_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Bc Fluxbp For Momentum_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin (Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
Name	“string”	–
plugin_parameters	“string”	–

Solid

Capillary

Syntax

Bc Fluxbp For Solid ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Capillary ()

Summary

Solid capillary flux BP BC

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–

Constant

Syntax

Bc Fluxbp For Solid ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Valuex = *valuex*] | [Valuey = *valuey*] | [Valuez = *valuez*])

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
valuex	real	0
valuey	real	0
valuez	real	0

Vector_String_Function

Syntax

Bc Fluxbp For Solid ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Vector_String_Function (F_X = *f_x* | F_Y = *f_y* | [F_Z = *f_z*])

Summary

Value from a string function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
f_x	“string”	–
f_y	“string”	–
f_z	“string”	–

Solid_X

Constant

Syntax

Bc Fluxbp For Solid_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant Value = *value*

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
value	real	–

Exponential

Syntax

Bc Fluxbp For Solid_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
variable	“string”	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Fluxbp For Solid_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global (Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Bc Fluxbp For Solid_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc Fluxbp For Solid_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Fluxbp For Solid_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function F = *f*

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
f	"string"	–

User_Field

Syntax

Bc Fluxbp For Solid_X ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc Fluxbp For Solid_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Bc Fluxbp For Solid_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin (Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
Name	“string”	–
plugin_parameters	“string”	–

Solid_Y

Constant

Syntax

Bc Fluxbp For Solid_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant Value = *value*

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
value	real	–

Exponential

Syntax

Bc Fluxbp For Solid_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
variable	“string”	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Fluxbp For Solid_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global (Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Bc Fluxbp For Solid_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc Fluxbp For Solid_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Fluxbp For Solid_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function F = *f*

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
f	"string"	–

User_Field

Syntax

Bc Fluxbp For Solid_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc Fluxbp For Solid_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Bc Fluxbp For Solid_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin (Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
Name	“string”	–
plugin_parameters	“string”	–

Solid_Z

Constant

Syntax

Bc Fluxbp For Solid_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant Value = *value*

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
value	real	–

Exponential

Syntax

Bc Fluxbp For Solid_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
variable	“string”	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Fluxbp For Solid_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global (Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Bc Fluxbp For Solid_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc Fluxbp For Solid_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Fluxbp For Solid_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function F = *f*

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
f	"string"	–

User_Field

Syntax

Bc Fluxbp For Solid_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc Fluxbp For Solid_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Bc Fluxbp For Solid_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin (Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
Name	“string”	–
plugin_parameters	“string”	–

Species

Surface_Stabilization

Syntax

Bc Fluxbp For Species ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Surface_Stabilization D = *d*

Summary

surface diffusion term for species

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
d	real	–

Bc Interface

Current

Butler_Volmer

Syntax

Bc Interface For Current ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Butler_Volmer ([Toggle = *toggle*] | F = *f* | R = *r* | Alpha_A = *alpha_a* | Alpha_C = *alpha_c* | [Flux_Lim_A = *flux_lim_a*] | [Flux_Lim_C = *flux_lim_c*] | Sign = *sign* | [Use_Solid_Stress = *use_solid_stress*] | [Use_Liquid_Stress = *use_liquid_stress*] | [Trans = *trans*] | [Reaction_Name = *reaction_name*])

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
f	real	–
r	real	–
alpha_a	real	–
alpha_c	real	–
flux_lim_a	real	0
flux_lim_c	real	0
sign	integer	–
use_solid_stress	“string”	–
use_liquid_stress	“string”	–
trans	real	0.3
reaction_name	“string”	–

Scalar_String_Function

Syntax

Bc Interface For Current ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
f	“string”	–

Cvfem_Continuity

Porous_Fluid_Coupling

Syntax

Bc Interface For Cvfem_Continuity ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) =

Porous_Fluid_Coupling ([Coeff_Scaling = *coeff_scaling*] | [Scale_Factor = *scale_factor*] | [Include_Average_Pressure = *include_average_pressure*] | [Multiplier = *multiplier*])

Summary

Interface BC for porous fluid coupling

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
coeff_scaling	real	1
scale_factor	real	1
include_average_pressure	integer	0
multiplier	real	1

Scalar_String_Function

Syntax

Bc Interface For Cvfem_Continuity ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function F = *f*

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
f	"string"	–

Cvfem_Energy

Porous_Fluid_Coupling

Syntax

Bc Interface For Cvfem_Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Porous_Fluid_Coupling ([Toggle = *toggle*] | [Coeff_Scaling = *coeff_scaling*] | [Scale_Factor = *scale_factor*] | [Include_Average_Pressure = *include_average_pressure*] | [Multiplier = *multiplier*])

Summary

Interface BC for porous fluid coupling

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
coeff_scaling	real	1
scale_factor	real	1
include_average_pressure	integer	0
multiplier	real	1

Porous_Fluid_Coupling_With_Solid_Convection

Syntax

Bc Interface For Cvfem_Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Porous_Fluid_Coupling_With_Solid_Convection ([Toggle = *toggle*] | [Coeff_Scaling = *coeff_scaling*] | [Scale_Factor = *scale_factor*] | [Include_Average_Pressure = *include_average_pressure*] | [Multiplier = *multiplier*])

Summary

Interface BC for porous fluid coupling

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
coeff_scaling	real	1
scale_factor	real	1
include_average_pressure	integer	0
multiplier	real	1

Scalar_String_Function

Syntax

Bc Interface For Cvfem_Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
f	“string”	–

Cvfem_Mass_Fraction

Porous_Fluid_Coupling

Syntax

Bc Interface For Cvfem_Mass_Fraction ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Porous_Fluid_Coupling ([Coeff_Scaling = *coeff_scaling*] | [Scale_Factor = *scale_factor*] | [Include_Average_Pressure = *include_average_pressure*] | [Multiplier = *multiplier*])

Summary

Interface BC for porous fluid coupling

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
coeff_scaling	real	1
scale_factor	real	1
include_average_pressure	integer	0
multiplier	real	1

Scalar_String_Function

Syntax

Bc Interface For Cvfem_Mass_Fraction ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function F = *f*

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
f	“string”	–

Energy

Cht_Robin

Syntax

Bc Interface For Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Cht_Robin ([Toggle = *toggle*] | [Opposite_Material_Phase = *opposite_material_phase*] | Opposite_Flux_Name = *opposite_flux_name* | [Coeff_Scaling = *coeff_scaling*])

Summary

Applies a Robin-style BC for the energy equation: $F_{local} = F_{bc} + \alpha(T_{local} - T_{bc})$.

Description

The following options are present in most boundary conditions/sources. The

meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
opposite_material_phase	"string"	–
opposite_flux_name	"string"	–
coeff_scaling	real	0.001

Contact_Resistance

Syntax

Bc Interface For Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Contact_Resistance ([Toggle = *toggle*] | [Opposite_Material_Phase = *opposite_material_phase*])

Summary

Applies a contact heat flux of the form $h(T - T_{opp})$ where h is the gap conductance coefficient defined for the surface material and the material phase of T_{opp} is optionally specified.

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
opposite_material_phase	"string"	–

Phase_Change

Syntax

Bc Interface For Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Phase_Change [Toggle = *toggle*]

Summary

Applies a energy flux to an interface based on phase change, as $q = v\rho_s(T_s C_{p,s} - T_g C_{p,g} + L)$ to sides s and g of the contact interface where v is the velocity ρ is the interface density C_p is the specific heat and L is the latent heat of fusion.

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–

Scalar_String_Function

Syntax

Bc Interface For Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
f	“string”	–

Lubrication_Height

Deforming

Syntax

Bc Interface For Lubrication_Height ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent*

Name ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Deforming [Hmin = *hmin*]

Summary

Lubrication height that deforms by the displacement of the adjoining volume region.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
hmin	real	0

Scalar_String_Function

Syntax

Bc Interface For Lubrication_Height ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function F = *f*

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
f	"string"	–

Mass_Balance

Porous_Fluid_Coupling

Syntax

Bc Interface For Mass_Balance ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Porous_Fluid_Coupling ([Coeff_Scaling = *coeff_scaling*] | [Scale_Factor = *scale_factor*] | [Include_Average_Pressure = *include_average_pressure*] | [Multiplier = *multiplier*])

Summary

Interface BC for porous fluid coupling

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
coeff_scaling	real	1
scale_factor	real	1
include_average_pressure	integer	0
multiplier	real	1

Scalar_String_Function

Syntax

Bc Interface For Mass_Balance ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function F = *f*

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
f	“string”	–

Species_Porous_Fluid_Coupling

Syntax

Bc Interface For Mass_Balance ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Species_Porous_Fluid_Coupling ([Coeff_Scaling = *coeff_scaling*] | [Scale_Factor = *scale_factor*] | [Include_Average_Pressure = *include_average_pressure*] | [Multiplier = *multiplier*])

Summary

Interface BC for porous fluid coupling

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
coeff_scaling	real	1
scale_factor	real	1
include_average_pressure	integer	0
multiplier	real	1

Mesh

Lubrication_Pressure

Syntax

Bc Interface For Mesh ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Lubrication_Pressure ()

Summary

Lubrication pressure mesh flux

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–

Momentum

Vector_String_Function

Syntax

Bc Interface For Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Vector_String_Function (F_X = *f_x* | F_Y = *f_y* | [F_Z = *f_z*])

Summary

Value from a string function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
f_x	“string”	–
f_y	“string”	–
f_z	“string”	–

Momentum_X

Scalar_String_Function

Syntax

Bc Interface For Momentum_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function F = *f*

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
f	"string"	–

Momentum_Y

Scalar_String_Function

Syntax

Bc Interface For Momentum_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function F = *f*

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
f	"string"	–

Momentum_Z

Scalar_String_Function

Syntax

Bc Interface For Momentum_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function F = *f*

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
f	“string”	–

Porous_Enthalpy

Contact_Resistance

Syntax

Bc Interface For Porous_Enthalpy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Contact_Resistance ([Opposite_Material_Phase = *opposite_material_phase*] | [Bulk_Node = *bulk_node*])

Summary

Applies a heat flux of the form $J \cdot n = h(T - T_{opp})$ where h is the gap

conductance coefficient defined for the surface material and the material phase of T_{opp} is optionally specified.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
opposite_material_phase	"string"	–
bulk_node	"string"	–

Fluid_Coupling_Solid_Convection

Syntax

Bc Interface For Porous_Enthalpy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Fluid_Coupling_Solid_Convection ([Coeff_Scaling = *coeff_scaling*] | [Scale_Factor = *scale_factor*] | [Include_Average_Pressure = *include_average_pressure*] | [Multiplier = *multiplier*])

Summary

Interface BC for porous fluid coupling

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
coeff_scaling	real	1
scale_factor	real	1
include_average_pressure	integer	0
multiplier	real	1

Porous_Fluid_Coupling

Syntax

Bc Interface For Porous_Enthalpy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Porous_Fluid_Coupling ([Coeff_Scaling = *coeff_scaling*] | [Scale_Factor = *scale_factor*] | [Include_Average_Pressure = *include_average_pressure*] | [Multiplier = *multiplier*])

Summary

Interface BC for porous fluid coupling

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
coeff_scaling	real	1
scale_factor	real	1
include_average_pressure	integer	0
multiplier	real	1

Porous_Fluid_Coupling_With_Solid_Convection

Syntax

Bc Interface For Porous_Enthalpy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Porous_Fluid_Coupling_With_Solid_Convection ([Coeff_Scaling = *coeff_scaling*] | [Scale_Factor = *scale_factor*] | [Include_Average_Pressure = *include_average_pressure*] | [Multiplier = *multiplier*])

Summary

Interface BC for porous fluid coupling

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
coeff_scaling	real	1
scale_factor	real	1
include_average_pressure	integer	0
multiplier	real	1

Scalar_String_Function

Syntax

Bc Interface For Porous_Enthalpy ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function F = *f*

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
f	“string”	–

Porous_Species

Butler_Volmer

Syntax

Bc Interface For Porous_Species ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Butler_Volmer (F = *f* | R = *r* | Alpha_A = *alpha_a* | Alpha_C = *alpha_c* | [Flux_Lim_A = *flux_lim_a*] | [Flux_Lim_C = *flux_lim_c*] | Sign = *sign* | Stoich = *stoich* | [Use_Solid_Stress = *use_solid_stress*] | [Use_Liquid_Stress = *use_liquid_stress*] | [Trans = *trans*] | [Reaction_Name = *reaction_name*])

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
f	real	–
r	real	–
alpha_a	real	–
alpha_c	real	–
flux_lim_a	real	0
flux_lim_c	real	0
sign	integer	–
stoich	real	–
use_solid_stress	“string”	–
use_liquid_stress	“string”	–
trans	real	0.3
reaction_name	“string”	–

Scalar_String_Function

Syntax

Bc Interface For Porous_Species ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function F = *f*

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
f	“string”	–

Species

Butler_Volmer

Syntax

Bc Interface For Species ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Butler_Volmer (F = *f* | R = *r* | Alpha_A = *alpha_a* | Alpha_C = *alpha_c* | [Flux_Lim_A = *flux_lim_a*] | [Flux_Lim_C = *flux_lim_c*] | Sign = *sign* | Stoich = *stoich* | [Use_Solid_Stress = *use_solid_stress*] | [Use_Liquid_Stress = *use_liquid_stress*] | [Trans = *trans*] | [Reaction_Name = *reaction_name*])

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
f	real	–
r	real	–
alpha_a	real	–
alpha_c	real	–
flux_lim_a	real	0
flux_lim_c	real	0
sign	integer	–
stoich	real	–
use_solid_stress	"string"	–
use_liquid_stress	"string"	–
trans	real	0.3
reaction_name	"string"	–

Reversible

Syntax

Bc Interface For Species ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Reversible (K_F = *k_f* | K_R = *k_r* | [Jump = *jump*] | Co_Species = *co_species*)

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
k_f	real	–
k_r	real	–
jump	real	0
co_species	"string"	–

Scalar_String_Function

Syntax

Bc Interface For Species ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function F = *f*

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
f	"string"	–

Tied

Syntax

Bc Interface For Species ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Tied [Co_Field = *co_field*]

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
co_field	integer	0

Bc Interface_Colloc_Disting

Cvfem_Energy

Matched_Temperature

Syntax

Bc Interface_Colloc_Disting For Cvfem_Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Matched_Temperature ()

Summary

Matched temperature for CVFEM energy interface collocated model

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–

Energy

Tied_Scalar

Syntax

Bc Interface_Colloc_Disting For Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Tied_Scalar ([Current_Expr = *current_expr*] | [Matched_Flux = *matched_flux*] | [Opposite_Expr = *opposite_expr*])

Summary

TIED_SCALAR for energy interface model

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
current_expr	“string”	–
matched_flux	“string”	–
opposite_expr	“string”	–

Lubrication

Tied_Pressure

Syntax

Bc Interface_Colloc_Disting For Lubrication ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Tied_Pressure [Matched_Flux = *matched_flux*]

Summary

TIED_PRESSURE for lubrication interface model

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
matched_flux	“string”	–

Tied_Scalar

Syntax

Bc Interface_Colloc_Disting For Lubrication ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Tied_Scalar ([Current_Expr = *current_expr*] | [Matched_Flux = *matched_flux*] | [Opposite_Expr = *opposite_expr*])

Summary

TIED_SCALAR for lubrication interface model

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
current_expr	"string"	–
matched_flux	"string"	–
opposite_expr	"string"	–

Lubrication_Height

Deforming

Syntax

Bc Interface_Colloc_Disting For Lubrication_Height ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Deforming [Hmin = *hmin*]

Summary

Collocated distinguishing condition for the lubrication height that deforms by the displacement of the adjoining volume region.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
hmin	real	0

Mass_Balance

Tied_Capillary_Pressure

Syntax

Bc Interface_Colloc_Disting For Mass_Balance ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Tied_Capillary_Pressure ([Wrt = *wrt*] | [Matched_Flux = *matched_flux*])

Summary

TIED_CAPILLARY_PRESSURE for MASS_BALANCE interface model

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
wrt	“string”	–
matched_flux	“string”	–

Mesh

Tied_Displacements

Syntax

Bc Interface_Colloc_Disting For Mesh ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Tied_Displacements ()

Summary

TIED_DISPLACEMENTS INTERFACE_COLLOC_DISTING MODEL FOR MESH

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–

Momentum

No_Slip

Syntax

Bc Interface_Colloc_Disting For Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = No_Slip [Matched_Flux = *matched_flux*]

Summary

NO_SLIP INTERFACE_COLLOC_DISTING MODEL FOR MOMENTUM

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
matched_flux	“string”	–

No_Slip_Solid_Velocity

Syntax

Bc Interface_Colloc_Disting For Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = No_Slip_Solid_Velocity [Matched_Flux = *matched_flux*]

Summary

NO_SLIP_SOLID_VELOCITY INTERFACE_COLLOC_DISTING MODEL FOR MOMENTUM

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
matched_flux	“string”	–

Tied_Vector

Syntax

Bc Interface_Colloc_Disting For Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Tied_Vector ([Current_Expr = *current_expr*] | [Matched_Flux = *matched_flux*] | [Opposite_Expr = *opposite_expr*])

Summary

TIED_VECTOR for momentum interface model

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
current_expr	"string"	–
matched_flux	"string"	–
opposite_expr	"string"	–

Porous_Enthalpy

Matched_Temperature

Syntax

Bc Interface_Colloc_Disting For Porous_Enthalpy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Matched_Temperature ()

Summary

Matched temperature for porous enthalpy interface collocated model

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–

Bc Interface_Disting

Cvfem_Energy

Matched_Temperature

Syntax

Bc Interface_Disting For Cvfem_Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Matched_Temperature ()

Summary

Matched temperature for CVFEM energy interface model

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–

Cvfem_Momentum

Porous_Fluid_Coupling

Syntax

Bc Interface_Disting For Cvfem_Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Porous_Fluid_Coupling [Alpha = *alpha*]

Summary

POROUS_FLUID_COUPLING INTERFACE_DISTING MODEL FOR CVFEM_MOMENTUM

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
alpha	real	0.1

Extension_Speed

Blowing

Syntax

Bc Interface_Disting For Extension_Speed ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Blowing ([Gamma = *gamma*] | [Ignore_Ale_Mesh_Motion = *ignore_ale_mesh_motion*])

Summary

Blowing interface disting BC model for extension speed

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
gamma	real	0
ignore_ale_mesh_motion	integer	–

Mesh

Recession

Syntax

Bc Interface_Disting For Mesh ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Recession ([P_Ref = *p_ref*] | K = *k* | K_0 = *k_0* | N = *n* | [Pressure = *pressure*])

Summary

RECESSION INTERFACE_COLLOC_DISTING MODEL FOR MESH

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
p_ref	real	0
k	real	–
k_0	real	–
n	real	–
pressure	“string”	–

Momentum

Blowing_No_Slip

Syntax

Bc Interface_Disting For Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Blowing_No_Slip ([Velocity = *velocity*] | [Opposing_Velocity = *opposing_velocity*])

Summary

BLOWING_NO_SLIP INTERFACE_COLLOC_DISTING MODEL FOR MOMENTUM

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
velocity	“string”	–
opposing_velocity	“string”	–

Porous_Enthalpy

Matched_Temperature

Syntax

Bc Interface_Disting For Porous_Enthalpy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Matched_Temperature ()

Summary

Matched temperature for porous enthalpy interface model

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–

Bc Lumped_Disting

Current

Constant

Syntax

Bc Lumped_Disting For Current ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant Value = *value*

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
value	real	–

Exponential

Syntax

Bc Lumped_Disting For Current ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Lumped_Disting For Current ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global (Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Bc Lumped_Disting For Current ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc Lumped_Disting For Current ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Lumped_Disting For Current ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function F = *f*

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about

valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
f	"string"	–

User_Field

Syntax

Bc Lumped_Disting For Current ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc Lumped_Disting For Current ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Bc Lumped_Disting For Current ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin (Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
Name	“string”	–
plugin_parameters	“string”	–

Cvfem_Energy

Constant

Syntax

Bc Lumped_Disting For Cvfem_Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant Value = *value*

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
value	real	–

Exponential

Syntax

Bc Lumped_Disting For Cvfem_Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
variable	“string”	–
constant	real	0
multiplier	real	1
exponent	real	–

From_Cht_Temperature

Syntax

Bc Lumped_Disting For Cvfem_Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = From_Cht_Temperature ([Twall_Field = *Twall_Field*] | [Poffset = *poffset*])

Summary

CVFEM energy distinguishing condition from CHT temperature using Cantera

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
Twall_Field	"string"	nodal_cht_Tw
poffset	real	0

From_Cht_Temperature_Cpt

Syntax

Bc Lumped_Disting For Cvfem_Energy ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = From_Cht_Temperature_Cpt [Twall_Field = *Twall_Field*]

Summary

CVFEM energy distinguishing condition from CHT temperature using Cp*T model

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
Twall_Field	"string"	nodal_cht_Tw

From_Temperature

Syntax

Bc Lumped_Disting For Cvfem_Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = From_Temperature (*{t | value}* = *t* | [Poffset = *poffset*])

Summary

From temperature distinguishing condition for CVFEM energy equation

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
t	real	–
poffset	real	0

Global

Syntax

Bc Lumped_Disting For Cvfem_Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global (Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Bc Lumped_Disting For Cvfem_Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc Lumped_Disting For Cvfem_Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Lumped_Disting For Cvfem_Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function F = *f*

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
f	"string"	–

User_Field

Syntax

Bc Lumped_Disting For Cvfem_Energy ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc Lumped_Disting For Cvfem_Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Bc Lumped_Disting For Cvfem_Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin (Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
Name	“string”	–
plugin_parameters	“string”	–

Cvfem_Level_Set

Constant

Syntax

Bc Lumped_Disting For Cvfem_Level_Set ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant Value = *value*

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
value	real	–

Exponential

Syntax

Bc Lumped_Disting For Cvfem_Level_Set ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
variable	“string”	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Lumped_Disting For Cvfem_Level_Set ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global (Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Bc Lumped_Disting For Cvfem_Level_Set ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc Lumped_Disting For Cvfem_Level_Set ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Lumped_Disting For Cvfem_Level_Set ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function F = *f*

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
f	"string"	–

User_Field

Syntax

Bc Lumped_Disting For Cvfem_Level_Set ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc Lumped_Disting For Cvfem_Level_Set ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Bc Lumped_Disting For Cvfem_Level_Set ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin (Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
Name	“string”	–
plugin_parameters	“string”	–

Cvfem_Momentum

No_Slip

Syntax

Bc Lumped_Disting For Cvfem_Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = No_Slip ()

Summary

No slip model for momentum distinguishing condition

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–

Porous_Robin_Interface

Syntax

Bc Lumped_Disting For Cvfem_Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Porous_Robin_Interface ([Alpha = *alpha*] | Permeability_Field = *permeability_field*)

Summary

No slip model for momentum distinguishing condition

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
alpha	real	0.1
permeability_field	“string”	–

Porous_Robin_One_Region

Syntax

Bc Lumped_Disting For Cvfem_Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Porous_Robin_One_Region [Alpha = *alpha*]

Summary

No slip model for momentum distinguishing condition

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
alpha	real	0.1

Cvfem_Specific_Dissipation_Rate

Constant

Syntax

Bc Lumped_Disting For Cvfem_Specific_Dissipation_Rate ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant Value = *value*

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
value	real	–

Exponential

Syntax

Bc Lumped_Disting For Cvfem_Specific_Dissipation_Rate ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
variable	“string”	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Lumped_Disting For Cvfem_Specific_Dissipation_Rate ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global (Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
global_name	"string"	–
component	integer	0

Low_Reynolds

Syntax

Bc Lumped_Disting For Cvfem_Specific_Dissipation_Rate ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Low_Reynolds ()

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–

Nd_Table

Syntax

Bc Lumped_Disting For Cvfem_Specific_Dissipation_Rate ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc Lumped_Disting For Cvfem_Specific_Dissipation_Rate ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Lumped_Disting For Cvfem_Specific_Dissipation_Rate ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function F = *f*

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
f	"string"	–

User_Field

Syntax

Bc Lumped_Disting For Cvfem_Specific_Dissipation_Rate ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc Lumped_Disting For Cvfem_Specific_Dissipation_Rate ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Bc Lumped_Disting For Cvfem_Specific_Dissipation_Rate ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin (Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
Name	"string"	–
plugin_parameters	"string"	–

Wall_Function

Syntax

Bc Lumped_Disting For Cvfem_Specific_Dissipation_Rate ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Wall_Function ([Wall_Friction_Factor = *Wall_Friction_Factor*] | [Wall_Velocity_X = *Wall_Velocity_X*] | [Wall_Velocity_Y = *Wall_Velocity_Y*] | [Wall_Velocity_Z = *Wall_Velocity_Z*])

Summary

CVFEM specific dissipation rate distinguishing condition

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
Wall_Friction_Factor	real	9.8
Wall_Velocity_X	real	0
Wall_Velocity_Y	real	0
Wall_Velocity_Z	real	0

Cvfem_Turbulence_Dissipation_Rate

Wall_Function

Syntax

Bc Lumped_Disting For Cvfem_Turbulence_Dissipation_Rate ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Wall_Function ([Wall_Friction_Factor = *Wall_Friction_Factor*] | [Wall_Velocity_X = *Wall_Velocity_X*] | [Wall_Velocity_Y = *Wall_Velocity_Y*] | [Wall_Velocity_Z = *Wall_Velocity_Z*])

Summary

CVFEM wall function turbulence dissipation rate distinguishing condition

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
Wall_Friction_Factor	real	9.8
Wall_Velocity_X	real	0
Wall_Velocity_Y	real	0
Wall_Velocity_Z	real	0

Cvfem_Turbulent_Kinetic_Energy

Wall_Function

Syntax

Bc Lumped_Disting For Cvfem_Turbulent_Kinetic_Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Wall_Function ([Wall_Friction_Factor = *Wall_Friction_Factor*] | [Wall_Velocity_X = *Wall_Velocity_X*] | [Wall_Velocity_Y = *Wall_Velocity_Y*] | [Wall_Velocity_Z = *Wall_Velocity_Z*])

Summary

CVFEM wall function turbulent kinetic energy distinguishing condition

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
Wall_Friction_Factor	real	9.8
Wall_Velocity_X	real	0
Wall_Velocity_Y	real	0
Wall_Velocity_Z	real	0

Energy

Constant

Syntax

Bc Lumped_Disting For Energy ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant Value = *value*

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
value	real	–

Exponential

Syntax

Bc Lumped_Disting For Energy ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
variable	“string”	–
constant	real	0
multiplier	real	1
exponent	real	–

From_Cht_Temperature

Syntax

Bc Lumped_Disting For Energy ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = From_Cht_Temperature (Interface_Name = *Interface_Name* | Twall_Field = *Twall_Field* | Tref_Field = *Tref_Field* | H_Field = *H_Field*)

Summary

Energy distinguishing condition from conjugate heat transfer temperature

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
Interface_Name	"string"	–
Twall_Field	"string"	–
Tref_Field	"string"	–
H_Field	"string"	–

Global

Syntax

Bc Lumped_Disting For Energy ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global (Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Bc Lumped_Disting For Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc Lumped_Disting For Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Lumped_Disting For Energy ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function F = *f*

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about

valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
f	"string"	–

User_Field

Syntax

Bc Lumped_Disting For Energy ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc Lumped_Disting For Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Bc Lumped_Disting For Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin (Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
Name	“string”	–
plugin_parameters	“string”	–

Enthalpy

Constant

Syntax

Bc Lumped_Disting For Enthalpy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant Value = *value*

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
value	real	–

Exponential

Syntax

Bc Lumped_Disting For Enthalpy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
variable	“string”	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Lumped_Disting For Enthalpy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global (Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Bc Lumped_Disting For Enthalpy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc Lumped_Disting For Enthalpy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Lumped_Disting For Enthalpy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function F = *f*

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
f	"string"	–

User_Field

Syntax

Bc Lumped_Disting For Enthalpy ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc Lumped_Disting For Enthalpy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Bc Lumped_Disting For Enthalpy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin (Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
Name	“string”	–
plugin_parameters	“string”	–

Extension_Speed

Advection

Syntax

Bc Lumped_Disting For Extension_Speed ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Advection ()

Summary

This model has been deprecated and removed from the code.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–

Constant

Syntax

Bc Lumped_Disting For Extension_Speed ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant Value = *value*

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
value	real	–

Exponential

Syntax

Bc Lumped_Disting For Extension_Speed ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Lumped_Disting For Extension_Speed ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global (Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
global_name	"string"	–
component	integer	0

Lens_Deposition

Syntax

Bc Lumped_Disting For Extension_Speed ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Lens_Deposition (Src_X = *src_x* | Dir_X = *dir_x* | Vel_X = *vel_x* | Src_Y = *src_y* | Dir_Y = *dir_y* | Vel_Y = *vel_y* | [Src_Z = *src_z*] | [Dir_Z = *dir_z*] | [Vel_Z = *vel_z*] | Massflowrate = *massFlowRate* | T_Liq = *T_liq* | T_Solid = *T_solid* | [T_On = *t_on*] | [T_Off = *t_off*] | Radius = *radius* | Particledensity = *particleDensity*)

Summary

LENS Deposition mass source model for extension speed distinguishing condition

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
src_x	real	–
dir_x	real	–
vel_x	real	–
src_y	real	–
dir_y	real	–
vel_y	real	–
src_z	real	0
dir_z	real	0
vel_z	real	0
massFlowRate	real	–
T_liq	real	–
T_solid	real	–
t_on	real	0
t_off	real	1.798e+308
radius	real	–
particleDensity	real	–

Melting

Syntax

Bc Lumped_Disting For Extension_Speed ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Melting ()

Summary

Melting model for extension speed distinguishing condition

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–

Nd_Table

Syntax

Bc Lumped_Disting For Extension_Speed ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc Lumped_Disting For Extension_Speed ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Pyrolysis

Syntax

Bc Lumped_Disting For Extension_Speed ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Pyrolysis ([T_Ref = *t_ref*] | K = *k* | K_0 = *k_0* | N = *n* | [Temperature = *temperature*])

Summary

Pyrolysis model for extension speed distinguishing condition

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
t_ref	real	0
k	real	–
k_0	real	–
n	real	–
temperature	“string”	–

Recession

Syntax

Bc Lumped_Disting For Extension_Speed ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Recession ([P_Ref = *p_ref*] | K = *k* | K_0 = *k_0* | N = *n* | [Pressure = *pressure*])

Summary

Advection model for extension speed distinguishing condition

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
p_ref	real	0
k	real	–
k_0	real	–
n	real	–
pressure	“string”	–

Scalar_String_Function

Syntax

Bc Lumped_Disting For Extension_Speed ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function F = *f*

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
f	“string”	–

Solidification

Syntax

Bc Lumped_Disting For Extension_Speed ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Solidification ()

Summary

Solidification model for extension speed distinguishing condition

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–

User_Field

Syntax

Bc Lumped_Disting For Extension_Speed ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
name	“string”	–
scaling	real	1
global_var	“string”	–
time_function	“string”	–

User_Function

Syntax

Bc Lumped_Disting For Extension_Speed ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Bc Lumped_Disting For Extension_Speed ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin (Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
Name	“string”	–
plugin_parameters	“string”	–

Level_Set

Advection

Syntax

Bc Lumped_Disting For Level_Set ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Advection ()

Summary

This model has been deprecated and removed from the code.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–

Constant

Syntax

Bc Lumped_Disting For Level_Set ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant Value = *value*

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
value	real	–

Exponential

Syntax

Bc Lumped_Disting For Level_Set ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Lumped_Disting For Level_Set ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global (Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Bc Lumped_Disting For Level_Set ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Outflow

Syntax

Bc Lumped_Disting For Level_Set ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Outflow [V0 = *v0*]

Summary

Outflow model for level set distinguishing condition

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
v0	real	0

Polynomial

Syntax

Bc Lumped_Disting For Level_Set ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Recession

Syntax

Bc Lumped_Disting For Level_Set ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Recession ([P_Ref = *p_ref*] | K = *k* | K_0 = *k_0* | N = *n* | [Pressure = *pressure*])

Summary

This model has been deprecated and removed from the code.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
p_ref	real	0
k	real	–
k_0	real	–
n	real	–
pressure	"string"	–

Scalar_String_Function

Syntax

Bc Lumped_Disting For Level_Set ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function F = *f*

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
f	"string"	–

User_Field

Syntax

Bc Lumped_Disting For Level_Set ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
name	“string”	–
scaling	real	1
global_var	“string”	–
time_function	“string”	–

User_Function

Syntax

Bc Lumped_Disting For Level_Set ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Bc Lumped_Disting For Level_Set ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin (Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
Name	“string”	–
plugin_parameters	“string”	–

Lubrication

Constant

Syntax

Bc Lumped_Disting For Lubrication ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant Value = *value*

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
value	real	–

Exponential

Syntax

Bc Lumped_Disting For Lubrication ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Lumped_Disting For Lubrication ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global (Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Bc Lumped_Disting For Lubrication ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc Lumped_Disting For Lubrication ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Lumped_Disting For Lubrication ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function F = *f*

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about

valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
f	"string"	–

User_Field

Syntax

Bc Lumped_Disting For Lubrication ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc Lumped_Disting For Lubrication ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Bc Lumped_Disting For Lubrication ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin (Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
Name	“string”	–
plugin_parameters	“string”	–

Mass_Balance

Constant

Syntax

Bc Lumped_Disting For Mass_Balance ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant Value = *value*

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
value	real	–

Exponential

Syntax

Bc Lumped_Disting For Mass_Balance ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
variable	“string”	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Lumped_Disting For Mass_Balance ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global (Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Bc Lumped_Disting For Mass_Balance ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc Lumped_Disting For Mass_Balance ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Lumped_Disting For Mass_Balance ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function F = *f*

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
f	"string"	–

User_Field

Syntax

Bc Lumped_Disting For Mass_Balance ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc Lumped_Disting For Mass_Balance ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Bc Lumped_Disting For Mass_Balance ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin (Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
Name	“string”	–
plugin_parameters	“string”	–

Mesh

Constant

Syntax

Bc Lumped_Disting For Mesh ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Valuex = *valuex*] | [Valuey = *valuey*] | [Valuez = *valuez*])

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
valuex	real	0
valuey	real	0
valuez	real	0

No_Slip

Syntax

Bc Lumped_Disting For Mesh ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = No_Slip [V0 = *v0*]

Summary

No slip model for mesh distinguishing condition

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
v0	real	0

Vector_String_Function

Syntax

Bc Lumped_Disting For Mesh ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Vector_String_Function (F_X = *f_x* | F_Y = *f_y* | [F_Z = *f_z*])

Summary

Value from a string function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
f_x	"string"	–
f_y	"string"	–
f_z	"string"	–

Mesh_X

Constant

Syntax

Bc Lumped_Disting For Mesh_X ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant Value = *value*

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
value	real	–

Exponential

Syntax

Bc Lumped_Disting For Mesh_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
variable	“string”	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Lumped_Disting For Mesh_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global (Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
global_name	"string"	–
component	integer	0

Kinematic

Syntax

Bc Lumped_Disting For Mesh_X ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Kinematic ()

Summary

Kinematic model for distinguishing condition has been removed, email sierra-help@sandia.gov for help.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–

Nd_Table

Syntax

Bc Lumped_Disting For Mesh_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc Lumped_Disting For Mesh_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Pyrolysis

Syntax

Bc Lumped_Disting For Mesh_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Pyrolysis ()

Summary

Pyrolysis model for distinguishing condition has been removed, email sierra-help@sandia.gov for help.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–

Recession

Syntax

Bc Lumped_Disting For Mesh_X ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Recession ()

Summary

Recession model for distinguishing condition has been removed, email sierra-help@sandia.gov for help.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–

Scalar_String_Function

Syntax

Bc Lumped_Disting For Mesh_X ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function F = *f*

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
f	"string"	–

User_Field

Syntax

Bc Lumped_Disting For Mesh_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc Lumped_Disting For Mesh_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Bc Lumped_Disting For Mesh_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin (Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
Name	“string”	–
plugin_parameters	“string”	–

Mesh_Y

Constant

Syntax

Bc Lumped_Disting For Mesh_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant Value = *value*

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
value	real	–

Exponential

Syntax

Bc Lumped_Disting For Mesh_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
variable	“string”	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Lumped_Disting For Mesh_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global (Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
global_name	"string"	–
component	integer	0

Kinematic

Syntax

Bc Lumped_Disting For Mesh_Y ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Kinematic ()

Summary

Kinematic model for distinguishing condition has been removed, email sierra-help@sandia.gov for help.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–

Nd_Table

Syntax

Bc Lumped_Disting For Mesh_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc Lumped_Disting For Mesh_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Pyrolysis

Syntax

Bc Lumped_Disting For Mesh_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Pyrolysis ()

Summary

Pyrolysis model for distinguishing condition has been removed, email sierra-help@sandia.gov for help.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–

Recession

Syntax

Bc Lumped_Disting For Mesh_Y ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Recession ()

Summary

Recession model for distinguishing condition has been removed, email sierra-help@sandia.gov for help.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–

Scalar_String_Function

Syntax

Bc Lumped_Disting For Mesh_Y ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function F = *f*

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
f	"string"	–

User_Field

Syntax

Bc Lumped_Disting For Mesh_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc Lumped_Disting For Mesh_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Bc Lumped_Disting For Mesh_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin (Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
Name	"string"	–
plugin_parameters	"string"	–

Mesh_Z

Constant

Syntax

Bc Lumped_Disting For Mesh_Z ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant Value = *value*

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
value	real	–

Exponential

Syntax

Bc Lumped_Disting For Mesh_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
variable	“string”	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Lumped_Disting For Mesh_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global (Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
global_name	"string"	–
component	integer	0

Kinematic

Syntax

Bc Lumped_Disting For Mesh_Z ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Kinematic ()

Summary

Kinematic model for distinguishing condition has been removed, email sierra-help@sandia.gov for help.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–

Nd_Table

Syntax

Bc Lumped_Disting For Mesh_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc Lumped_Disting For Mesh_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Pyrolysis

Syntax

Bc Lumped_Disting For Mesh_Z ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Pyrolysis ()

Summary

Pyrolysis model for distinguishing condition has been removed, email sierra-help@sandia.gov for help.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–

Recession

Syntax

Bc Lumped_Disting For Mesh_Z ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Recession ()

Summary

Recession model for distinguishing condition has been removed, email sierra-help@sandia.gov for help.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–

Scalar_String_Function

Syntax

Bc Lumped_Disting For Mesh_Z ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function F = *f*

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
f	"string"	–

User_Field

Syntax

Bc Lumped_Disting For Mesh_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc Lumped_Disting For Mesh_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Bc Lumped_Disting For Mesh_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin (Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
Name	“string”	–
plugin_parameters	“string”	–

Momentum

Constant

Syntax

Bc Lumped_Disting For Momentum ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Valuex = *valuex*] | [Valuey = *valuey*] | [Valuez = *valuez*])

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
valuex	real	0
valuey	real	0
valuez	real	0

Darcy_Leak

Syntax

Bc Lumped_Disting For Momentum ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Darcy_Leak ([P_Ref = *p_ref*] | [L = *l*] | K = *k* | [Pressure = *pressure*])

Summary

Darcy leak model for momentum distinguishing condition

Description

This model implements a vector distinguishing condition that allows fluid to "leak" through a boundary according to a Darcy's Law like relationship:

$$\boldsymbol{v} - \dot{\boldsymbol{x}} \cdot \boldsymbol{n}\boldsymbol{n} - \frac{K}{L\mu}\left(P - P_{ref}\right)\boldsymbol{n} = \boldsymbol{0}.$$

Here $\boldsymbol{n}$ is the outward unit normal to the boundary, $\boldsymbol{v}$ is the velocity, $\dot{\boldsymbol{x}}$ is the time derivative of the mesh boundary position and P is the fluid pressure. K is the permeability of the boundary, μ is the viscosity, P_{ref} is a reference pressure (defaults to zero) and L is a length scale for the pressure drop (defaults to unity).

Note that the second term ensures that the resulting velocity is relative to any boundary motion that arises in moving mesh problems. For problems on fixed meshes this term is automatically excluded.

An alternate pressure field can be used by provided the optional parameter `PRESSURE`.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
p_ref	real	0
l	real	1
k	real	–
pressure	"string"	–

Electroosmotic_Velocity

Syntax

Bc Lumped_Disting For Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Electroosmotic_Velocity C = *c*

Summary

Electro-osmotic velocity model for momentum distinguishing condition

Description

This distinguishing condition sets the velocity to simply be a constant multiple of the electric field,

$$\boldsymbol{v} - c\boldsymbol{E} = \boldsymbol{0}$$

where $\boldsymbol{v}$ is the fluid velocity, $\boldsymbol{E}$ is the electric field and c is the constant C supplied by the user.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
c	real	–

No_Slip

Syntax

Bc Lumped_Disting For Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = No_Slip [V0 = *v0*]

Summary

No slip model for momentum distinguishing condition

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
v0	real	0

Vector_String_Function

Syntax

Bc Lumped_Disting For Momentum ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Vector_String_Function (F_X = *f_x* | F_Y = *f_y* | [F_Z = *f_z*])

Summary

Value from a string function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
f_x	“string”	–
f_y	“string”	–
f_z	“string”	–

Vector_User_Function_Disting

Syntax

Bc Lumped_Disting For Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Vector_User_Function_Disting ([Name_X = *name_x*] | [Name_Y = *name_y*] | [Name_Z = *name_z*] | X = *x* | [Multiplier = *multiplier*])

Summary

Vector user function model for momentum distinguishing condition

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
name_x	“string”	–
name_y	“string”	–
name_z	“string”	–
x	“string”	–
multiplier	real	1

Wetting_Speed_Blake_Ls

Syntax

Bc Lumped_Disting For Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Wetting_Speed_Blake_Ls (V_W = *v_w* | G = *g* | Theta = *theta* | Width = *width* | [Tau = *tau*])

Summary

Blake level set wetting speed model for momentum distinguishing condition

Description

The wetting condition is governed by

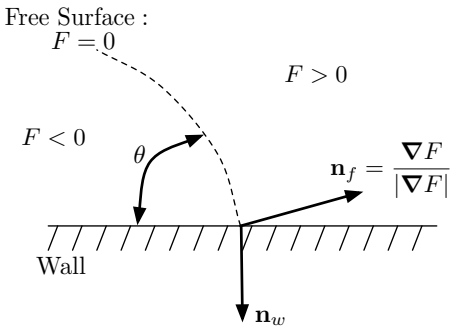
$$\boldsymbol{v} - \delta(F) v_w \sinh\left(g\left(\cos\theta_s - \cos\theta\right)\right)\boldsymbol{t}_w + \delta(F)\tau\dot{\boldsymbol{v}} = 0$$

where $\boldsymbol{v}$ is the fluid velocity and $\boldsymbol{t}_w$ is the tangent to the wall. The function $\delta(F)$, where F is the level set distance field, is given by

$$\delta(F) = \frac{1}{2}\left(1 + \cos\frac{\pi F}{\alpha}\right)$$

when $|F| < \alpha$ and zero elsewhere. Here, α is the half of the `WIDTH` parameter. The term involving τ is a transient relaxation term. By default, $\tau = 0$.

This distinguishing condition is a function of the static and observed contact angles. The static contact angle θ_s, supplied by the `THETA` parameter, is fixed. The observed contact angle θ is computed from the current state of the solution as illustrated in the following diagram. The important point here is that the contact angle is measured through the negative side of the distance function which is denoted `PHASE_A` in Aria.



Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
v_w	real	–
g	real	–
theta	real	–
width	real	–
tau	real	0

Momentum_X

Constant

Syntax

Bc Lumped_Disting For Momentum_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant Value = *value*

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
value	real	–

Exponential

Syntax

Bc Lumped_Disting For Momentum_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Lumped_Disting For Momentum_X ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global (Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Bc Lumped_Disting For Momentum_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc Lumped_Disting For Momentum_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Lumped_Disting For Momentum_X ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function F = *f*

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about

valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
f	"string"	–

User_Field

Syntax

Bc Lumped_Disting For Momentum_X ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc Lumped_Disting For Momentum_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Bc Lumped_Disting For Momentum_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin (Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
Name	“string”	–
plugin_parameters	“string”	–

Momentum_Y

Constant

Syntax

Bc Lumped_Disting For Momentum_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant Value = *value*

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
value	real	–

Exponential

Syntax

Bc Lumped_Disting For Momentum_Y ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
variable	“string”	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Lumped_Disting For Momentum_Y ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global (Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Bc Lumped_Disting For Momentum_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc Lumped_Disting For Momentum_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Lumped_Disting For Momentum_Y ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function F = *f*

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
f	"string"	–

User_Field

Syntax

Bc Lumped_Disting For Momentum_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc Lumped_Disting For Momentum_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Bc Lumped_Disting For Momentum_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin (Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
Name	“string”	–
plugin_parameters	“string”	–

Momentum_Z

Constant

Syntax

Bc Lumped_Disting For Momentum_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant Value = *value*

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
value	real	–

Exponential

Syntax

Bc Lumped_Disting For Momentum_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
variable	“string”	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Lumped_Disting For Momentum_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global (Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Bc Lumped_Disting For Momentum_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc Lumped_Disting For Momentum_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Lumped_Disting For Momentum_Z ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function F = *f*

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
f	"string"	–

User_Field

Syntax

Bc Lumped_Disting For Momentum_Z ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc Lumped_Disting For Momentum_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Bc Lumped_Disting For Momentum_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin (Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
Name	"string"	–
plugin_parameters	"string"	–

Porous_Enthalpy

Constant

Syntax

Bc Lumped_Disting For Porous_Enthalpy ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant Value = *value*

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
value	real	–

Exponential

Syntax

Bc Lumped_Disting For Porous_Enthalpy ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
variable	“string”	–
constant	real	0
multiplier	real	1
exponent	real	–

From_Cht_Temperature

Syntax

Bc Lumped_Disting For Porous_Enthalpy ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = From_Cht_Temperature (Interface_Name = *Interface_Name* | Twall_Field = *Twall_Field* | Tref_Field = *Tref_Field* | H_Field = *H_Field*)

Summary

Porous enthalpy distinguishing condition from conjugate heat transfer temperature

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
Interface_Name	“string”	–
Twall_Field	“string”	–
Tref_Field	“string”	–
H_Field	“string”	–

Global

Syntax

Bc Lumped_Disting For Porous_Enthalpy ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global (Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
global_name	“string”	–
component	integer	0

Nd_Table

Syntax

Bc Lumped_Disting For Porous_Enthalpy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc Lumped_Disting For Porous_Enthalpy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Lumped_Disting For Porous_Enthalpy ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function F = *f*

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about

valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
f	"string"	–

User_Field

Syntax

Bc Lumped_Disting For Porous_Enthalpy ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc Lumped_Disting For Porous_Enthalpy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Bc Lumped_Disting For Porous_Enthalpy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin (Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
Name	"string"	–
plugin_parameters	"string"	–

Solid

Constant

Syntax

Bc Lumped_Disting For Solid ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Valuex = *valuex*] | [Valuey = *valuey*] | [Valuez = *valuez*])

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
valuex	real	0
valuey	real	0
valuez	real	0

No_Slip

Syntax

Bc Lumped_Disting For Solid ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = No_Slip [V0 = *v0*]

Summary

No slip model for SOLID distinguishing condition

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
v0	real	0

Vector_String_Function

Syntax

Bc Lumped_Disting For Solid ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Vector_String_Function (F_X = *f_x* | F_Y = *f_y* | [F_Z = *f_z*])

Summary

Value from a string function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
f_x	“string”	–
f_y	“string”	–
f_z	“string”	–

Solid_X

Constant

Syntax

Bc Lumped_Disting For Solid_X ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant Value = *value*

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
value	real	–

Exponential

Syntax

Bc Lumped_Disting For Solid_X ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
variable	“string”	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Lumped_Disting For Solid_X ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global (Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
global_name	"string"	–
component	integer	0

Kinematic

Syntax

Bc Lumped_Disting For Solid_X ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Kinematic ()

Summary

Kinematic model for distinguishing condition. Will be deprecated in 4.59, email sierra-help@sandia.gov for help.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–

Nd_Table

Syntax

Bc Lumped_Disting For Solid_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc Lumped_Disting For Solid_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Pyrolysis

Syntax

Bc Lumped_Disting For Solid_X ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Pyrolysis ()

Summary

Pyrolysis model for distinguishing condition has been removed , email sierra-help@sandia.gov for help.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–

Recession

Syntax

Bc Lumped_Disting For Solid_X ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Recession ()

Summary

Recession model for distinguishing condition has been removed, email sierra-help@sandia.gov for help.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–

Scalar_String_Function

Syntax

Bc Lumped_Disting For Solid_X ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function F = *f*

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
f	"string"	–

User_Field

Syntax

Bc Lumped_Disting For Solid_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc Lumped_Disting For Solid_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Bc Lumped_Disting For Solid_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin (Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
Name	"string"	–
plugin_parameters	"string"	–

Solid_Y

Constant

Syntax

Bc Lumped_Disting For Solid_Y ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant Value = *value*

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
value	real	–

Exponential

Syntax

Bc Lumped_Disting For Solid_Y ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
variable	“string”	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Lumped_Disting For Solid_Y ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global (Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
global_name	"string"	–
component	integer	0

Kinematic

Syntax

Bc Lumped_Disting For Solid_Y ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Kinematic ()

Summary

Kinematic model for distinguishing condition. Will be deprecated in 4.59, email sierra-help@sandia.gov for help.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–

Nd_Table

Syntax

Bc Lumped_Disting For Solid_Y ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc Lumped_Disting For Solid_Y ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Pyrolysis

Syntax

Bc Lumped_Disting For Solid_Y ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Pyrolysis ()

Summary

Pyrolysis model for distinguishing condition has been removed , email sierra-help@sandia.gov for help.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–

Recession

Syntax

Bc Lumped_Disting For Solid_Y ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Recession ()

Summary

Recession model for distinguishing condition has been removed, email sierra-help@sandia.gov for help.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–

Scalar_String_Function

Syntax

Bc Lumped_Disting For Solid_Y ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function F = *f*

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
f	"string"	–

User_Field

Syntax

Bc Lumped_Disting For Solid_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc Lumped_Disting For Solid_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Bc Lumped_Disting For Solid_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin (Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
Name	“string”	–
plugin_parameters	“string”	–

Solid_Z

Constant

Syntax

Bc Lumped_Disting For Solid_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant Value = *value*

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
value	real	–

Exponential

Syntax

Bc Lumped_Disting For Solid_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
variable	“string”	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Lumped_Disting For Solid_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global (Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
global_name	"string"	–
component	integer	0

Kinematic

Syntax

Bc Lumped_Disting For Solid_Z ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Kinematic ()

Summary

Kinematic model for distinguishing condition. Will be deprecated in 4.59, email sierra-help@sandia.gov for help.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–

Nd_Table

Syntax

Bc Lumped_Disting For Solid_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc Lumped_Disting For Solid_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Pyrolysis

Syntax

Bc Lumped_Disting For Solid_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Pyrolysis ()

Summary

Pyrolysis model for distinguishing condition has been removed , email sierra-help@sandia.gov for help.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–

Recession

Syntax

Bc Lumped_Disting For Solid_Z ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Recession ()

Summary

Recession model for distinguishing condition has been removed, email sierra-help@sandia.gov for help.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–

Scalar_String_Function

Syntax

Bc Lumped_Disting For Solid_Z ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function F = *f*

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
f	"string"	–

User_Field

Syntax

Bc Lumped_Disting For Solid_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc Lumped_Disting For Solid_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Bc Lumped_Disting For Solid_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin (Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
Name	“string”	–
plugin_parameters	“string”	–

Species

Constant

Syntax

Bc Lumped_Disting For Species ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant Value = *value*

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
value	real	–

Exponential

Syntax

Bc Lumped_Disting For Species ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
variable	“string”	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Lumped_Disting For Species ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global (Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Bc Lumped_Disting For Species ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc Lumped_Disting For Species ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Lumped_Disting For Species ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function F = *f*

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
f	"string"	–

User_Field

Syntax

Bc Lumped_Disting For Species ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc Lumped_Disting For Species ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Bc Lumped_Disting For Species ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin (Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
Name	"string"	–
plugin_parameters	"string"	–

Voltage

Constant

Syntax

Bc Lumped_Disting For Voltage ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant Value = *value*

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
value	real	–

Exponential

Syntax

Bc Lumped_Disting For Voltage ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
variable	“string”	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Lumped_Disting For Voltage ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global (Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Bc Lumped_Disting For Voltage ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc Lumped_Disting For Voltage ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Lumped_Disting For Voltage ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function F = *f*

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
f	"string"	–

User_Field

Syntax

Bc Lumped_Disting For Voltage ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc Lumped_Disting For Voltage ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Bc Lumped_Disting For Voltage ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin (Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
Name	"string"	–
plugin_parameters	"string"	–

Bc Lumped_Flux

Ampere

Constant

Syntax

Bc Lumped_Flux For Ampere ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | [Valuex = *valuex*] | [Valuey = *valuey*] | [Valuez = *valuez*])

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
valuex	real	0
valuey	real	0
valuez	real	0

Vector_String_Function

Syntax

Bc Lumped_Flux For Ampere ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Vector_String_Function ([Toggle = *toggle*] | F_X = *f_x* | F_Y = *f_y* | [F_Z = *f_z*])

Summary

Value from a string function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f_x	"string"	–
f_y	"string"	–
f_z	"string"	–

Ampere_X

Constant

Syntax

Bc Lumped_Flux For Ampere_X ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | Value = *value*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
value	real	–

Exponential

Syntax

Bc Lumped_Flux For Ampere_X ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Lumped_Flux For Ampere_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
global_name	“string”	–
component	integer	0

Nd_Table

Syntax

Bc Lumped_Flux For Ampere_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc Lumped_Flux For Ampere_X ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Lumped_Flux For Ampere_X ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f	"string"	–

User_Field

Syntax

Bc Lumped_Flux For Ampere_X ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
scaling	real	1
global_var	“string”	–
time_function	“string”	–

User_Function

Syntax

Bc Lumped_Flux For Ampere_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc Lumped_Flux For Ampere_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

Ampere_Y

Constant

Syntax

Bc Lumped_Flux For Ampere_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | Value = *value*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
value	real	–

Exponential

Syntax

Bc Lumped_Flux For Ampere_Y ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Lumped_Flux For Ampere_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Bc Lumped_Flux For Ampere_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc Lumped_Flux For Ampere_Y ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Lumped_Flux For Ampere_Y ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f	"string"	–

User_Field

Syntax

Bc Lumped_Flux For Ampere_Y ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
scaling	real	1
global_var	“string”	–
time_function	“string”	–

User_Function

Syntax

Bc Lumped_Flux For Ampere_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc Lumped_Flux For Ampere_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

Ampere_Z

Constant

Syntax

Bc Lumped_Flux For Ampere_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | Value = *value*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
value	real	–

Exponential

Syntax

Bc Lumped_Flux For Ampere_Z ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Lumped_Flux For Ampere_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Bc Lumped_Flux For Ampere_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc Lumped_Flux For Ampere_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Lumped_Flux For Ampere_Z ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f	"string"	–

User_Field

Syntax

Bc Lumped_Flux For Ampere_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc Lumped_Flux For Ampere_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc Lumped_Flux For Ampere_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

Charge_Density

Constant

Syntax

Bc Lumped_Flux For Charge_Density ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | *{flux | value}* = *flux*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
flux	real	–

Exponential

Syntax

Bc Lumped_Flux For Charge_Density ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Lumped_Flux For Charge_Density ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
global_name	"string"	–
component	integer	0

Nat_E

Syntax

Bc Lumped_Flux For Charge_Density ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nat_E ([Toggle = *toggle*] | [Sigma = *sigma*])

Summary

This model is now deprecated

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
sigma	"string"	–

Nd_Table

Syntax

Bc Lumped_Flux For Charge_Density ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc Lumped_Flux For Charge_Density ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
variable	“string”	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Lumped_Flux For Charge_Density ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
f	“string”	–

User_Field

Syntax

Bc Lumped_Flux For Charge_Density ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc Lumped_Flux For Charge_Density ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc Lumped_Flux For Charge_Density ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

Continuity

Constant

Syntax

Bc Lumped_Flux For Continuity ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | Value = *value*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
value	real	–

Exponential

Syntax

Bc Lumped_Flux For Continuity ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Extrapolated_Pressure

Syntax

Bc Lumped_Flux For Continuity ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Extrapolated_Pressure [Toggle = *toggle*]

Summary

This model has been deprecated and removed from the code.

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–

Global

Syntax

Bc Lumped_Flux For Continuity ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
global_name	“string”	–
component	integer	0

Nd_Table

Syntax

Bc Lumped_Flux For Continuity ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc Lumped_Flux For Continuity ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
variable	“string”	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Pspg

Syntax

Bc Lumped_Flux For Continuity ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Pspg ([Toggle = *toggle*] | [Tausurfacefactor = *TauSurfaceFactor*] | [Tausurfacejacobianfactor = *TauSurfaceJacobianFactor*])

Summary

Continuity PSPG flux BC

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
TauSurfaceFactor	real	1
TauSurfaceJacobianFactor	real	1

Scalar_String_Function

Syntax

Bc Lumped_Flux For Continuity ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f	"string"	–

User_Field

Syntax

Bc Lumped_Flux For Continuity ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc Lumped_Flux For Continuity ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc Lumped_Flux For Continuity ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

Current

Calore_User_Sub

Syntax

Bc Lumped_Flux For Current ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Calore_User_Sub ([Toggle = *toggle*] | Name = *name* | Type = *type* | [Multiplier = *multiplier*] | [Material_Data_Block = *material_data_block*] | [Data = *data*] | [Scaling_Field = *scaling_field*])

Summary

Values from a Calore user subroutine

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
type	"string"	–
multiplier	real	1
material_data_block	"string"	–
data	"string"	–
scaling_field	"string"	–

Constant

Syntax

Bc Lumped_Flux For Current ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | *{flux | value}* = *flux*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
flux	real	–

Constant_With_Cutoff_Voltage

Syntax

Bc Lumped_Flux For Current ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant_With_Cutoff_Voltage ([Toggle = *toggle*] | Value = *value* | Cutoff_Value = *cutoff_value*)

Summary

Apply a constant current until a cutoff voltage is crossed anywhere on the surface. After that apply 0 current.

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
value	real	–
cutoff_value	real	–

Exponential

Syntax

Bc Lumped_Flux For Current ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Lumped_Flux For Current ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Bc Lumped_Flux For Current ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc Lumped_Flux For Current ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Resistive_Load

Syntax

Bc Lumped_Flux For Current ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Resistive_Load ([Toggle = *toggle*] | [Area = *Area*] | [Referencevoltage = *ReferenceVoltage*])

Summary

Model a boundary attached to a resistive load. The local applied current density is V / (R A) where V is the local voltage, R is the specified resistance, and A is the area of the surface the BC is applied to.

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Area	real	-1
ReferenceVoltage	real	0

Scalar_String_Function

Syntax

Bc Lumped_Flux For Current ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
f	“string”	–

User_Field

Syntax

Bc Lumped_Flux For Current ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc Lumped_Flux For Current ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc Lumped_Flux For Current ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

Cvfem_Continuity

Constant

Syntax

Bc Lumped_Flux For Cvfem_Continuity ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | Value = *value*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
value	real	–

Exponential

Syntax

Bc Lumped_Flux For Cvfem_Continuity ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Lumped_Flux For Cvfem_Continuity ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
global_name	"string"	–
component	integer	0

Inflow

Syntax

Bc Lumped_Flux For Cvfem_Continuity ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Inflow [Toggle = *toggle*]

Summary

IBP continuity inflow; nodal interp

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–

Interface

Syntax

Bc Lumped_Flux For Cvfem_Continuity ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Interface [Toggle = *toggle*]

Summary

INTERFACE CVFEM continuity flux

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–

Mass_Flux

Syntax

Bc Lumped_Flux For Cvfem_Continuity ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Mass_Flux ([Toggle = *toggle*] | Total_Mdot = *total_mdot*)

Summary

Mass flux CVFEM continuity flux

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
total_mdot	real	–

Nd_Table

Syntax

Bc Lumped_Flux For Cvfem_Continuity ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Open_Flow

Syntax

Bc Lumped_Flux For Cvfem_Continuity ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Open_Flow ([Toggle = *toggle*] | [Pressure = *pressure*] | [Total_Pressure = *total_pressure*])

Summary

Open bc for continuity

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
pressure	real	0
total_pressure	real	0

Polynomial

Syntax

Bc Lumped_Flux For Cvfem_Continuity ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
variable	“string”	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Porous_Robin_Averaged_Coeff

Syntax

Bc Lumped_Flux For Cvfem_Continuity ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Porous_Robin_Averaged_Coeff ([Toggle = *toggle*] | P_Field = *P_Field* | Massflux_Field = *MassFlux_Field* | Oppositecoeff_Field = *OppositeCoeff_Field*)

Summary

Boundary condition for loose coupling of continuity to a free fluid region

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
P_Field	"string"	–
MassFlux_Field	"string"	–
OppositeCoeff_Field	"string"	–

Porous_Robin_One_Region

Syntax

Bc Lumped_Flux For Cvfem_Continuity ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Porous_Robin_One_Region [Toggle = *toggle*]

Summary

Boundary condition for loose coupling of continuity to a free fluid region

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–

Scalar_String_Function

Syntax

Bc Lumped_Flux For Cvfem_Continuity ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
f	“string”	–

Simple_Inflow

Syntax

Bc Lumped_Flux For Cvfem_Continuity ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Simple_Inflow ([Toggle = *toggle*] | *{density | value}* = *Density* | [Velocity_X = *Velocity_X*] | [Velocity_Y = *Velocity_Y*] | [Velocity_Z = *Velocity_Z*])

Summary

IBP continuity inflow; specified dens and u

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Density	real	–
Velocity_X	real	0
Velocity_Y	real	0
Velocity_Z	real	0

User_Field

Syntax

Bc Lumped_Flux For Cvfem_Continuity ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc Lumped_Flux For Cvfem_Continuity ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc Lumped_Flux For Cvfem_Continuity ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

Cvfem_Density_Edge_Gradient_Projection

Constant

Syntax

Bc Lumped_Flux For Cvfem_Density_Edge_Gradient_Projection ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | Value = *value*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
value	real	–

Exponential

Syntax

Bc Lumped_Flux For Cvfem_Density_Edge_Gradient_Projection ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Lumped_Flux For Cvfem_Density_Edge_Gradient_Projection ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Bc Lumped_Flux For Cvfem_Density_Edge_Gradient_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc Lumped_Flux For Cvfem_Density_Edge_Gradient_Projection ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Lumped_Flux For Cvfem_Density_Edge_Gradient_Projection ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f	"string"	–

Simple_Interp

Syntax

Bc Lumped_Flux For Cvfem_Density_Edge_Gradient_Projection ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Simple_Interp [Toggle = *toggle*]

Summary

CVFEM edge-based projection simple interpolation flux BC

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–

User_Field

Syntax

Bc Lumped_Flux For Cvfem_Density_Edge_Gradient_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
scaling	real	1
global_var	“string”	–
time_function	“string”	–

User_Function

Syntax

Bc Lumped_Flux For Cvfem_Density_Edge_Gradient_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc Lumped_Flux For Cvfem_Density_Edge_Gradient_Projection ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

Cvfem_Dispersed_Continuity

Constant

Syntax

Bc Lumped_Flux For Cvfem_Dispersed_Continuity ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | Value = *value*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
value	real	–

Exponential

Syntax

Bc Lumped_Flux For Cvfem_Dispersed_Continuity ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Lumped_Flux For Cvfem_Dispersed_Continuity ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
global_name	"string"	–
component	integer	0

Inflow

Syntax

Bc Lumped_Flux For Cvfem_Dispersed_Continuity ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Inflow [Toggle = *toggle*]

Summary

Inflow mass flux for dispersed phase continuity equation; nodal interpolation

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–

Nd_Table

Syntax

Bc Lumped_Flux For Cvfem_Dispersed_Continuity ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Open_Flow

Syntax

Bc Lumped_Flux For Cvfem_Dispersed_Continuity ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Open_Flow [Toggle = *toggle*]

Summary

Open bc for dispersed continuity

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–

Polynomial

Syntax

Bc Lumped_Flux For Cvfem_Dispersed_Continuity ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset,

X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Lumped_Flux For Cvfem_Dispersed_Continuity ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be

a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f	"string"	–

Simple_Inflow

Syntax

Bc Lumped_Flux For Cvfem_Dispersed_Continuity ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Simple_Inflow ([Toggle = *toggle*] | [Velocity_X = *Velocity_X*] | [Velocity_Y = *Velocity_Y*] | [Velocity_Z = *Velocity_Z*])

Summary

Inflow mass flux for dispersed continuity equation; specified density and velocity

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Velocity_X	real	0
Velocity_Y	real	0
Velocity_Z	real	0

User_Field

Syntax

Bc Lumped_Flux For Cvfem_Dispersed_Continuity ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc Lumped_Flux For Cvfem_Dispersed_Continuity ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc Lumped_Flux For Cvfem_Dispersed_Continuity ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

Cvfem_Dispersed_Momentum

Constant

Syntax

Bc Lumped_Flux For Cvfem_Dispersed_Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | [Valuex = *valuex*] | [Valuey = *valuey*] | [Valuez = *valuez*])

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
valuex	real	0
valuey	real	0
valuez	real	0

Open_Adv_Flow

Syntax

Bc Lumped_Flux For Cvfem_Dispersed_Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Open_Adv_Flow ([Toggle = *toggle*] | [Coflow_Entrainment_Valuex = *Coflow_Entrainment_ValueX*] | [Coflow_Entrainment_Valuey = *Coflow_Entrainment_ValueY*] | [Coflow_Entrainment_Valuez = *Coflow_Entrainment_ValueZ*])

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Coflow_Entrainment_ValueX	real	0
Coflow_Entrainment_ValueY	real	0
Coflow_Entrainment_ValueZ	real	0

Open_Flow

Syntax

Bc Lumped_Flux For Cvfem_Dispersed_Momentum ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Open_Flow [Toggle = *toggle*]

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–

Symmetry_Flow

Syntax

Bc Lumped_Flux For Cvfem_Dispersed_Momentum ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Symmetry_Flow [Toggle = *toggle*]

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–

Vector_String_Function

Syntax

Bc Lumped_Flux For Cvfem_Dispersed_Momentum ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Vector_String_Function ([Toggle = *toggle*] | F_X = *f_x* | F_Y = *f_y* | [F_Z = *f_z*])

Summary

Value from a string function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f_x	"string"	–
f_y	"string"	–
f_z	"string"	–

Cvfem_Dispersed_Momentum_X

Constant

Syntax

Bc Lumped_Flux For Cvfem_Dispersed_Momentum_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | Value = *value*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
value	real	–

Exponential

Syntax

Bc Lumped_Flux For Cvfem_Dispersed_Momentum_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Lumped_Flux For Cvfem_Dispersed_Momentum_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Bc Lumped_Flux For Cvfem_Dispersed_Momentum_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc Lumped_Flux For Cvfem_Dispersed_Momentum_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Lumped_Flux For Cvfem_Dispersed_Momentum_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f	"string"	–

User_Field

Syntax

Bc Lumped_Flux For Cvfem_Dispersed_Momentum_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc Lumped_Flux For Cvfem_Dispersed_Momentum_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc Lumped_Flux For Cvfem_Dispersed_Momentum_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

Cvfem_Dispersed_Momentum_Y

Constant

Syntax

Bc Lumped_Flux For Cvfem_Dispersed_Momentum_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | Value = *value*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
value	real	–

Exponential

Syntax

Bc Lumped_Flux For Cvfem_Dispersed_Momentum_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Lumped_Flux For Cvfem_Dispersed_Momentum_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Bc Lumped_Flux For Cvfem_Dispersed_Momentum_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc Lumped_Flux For Cvfem_Dispersed_Momentum_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Lumped_Flux For Cvfem_Dispersed_Momentum_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f	"string"	–

User_Field

Syntax

Bc Lumped_Flux For Cvfem_Dispersed_Momentum_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc Lumped_Flux For Cvfem_Dispersed_Momentum_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc Lumped_Flux For Cvfem_Dispersed_Momentum_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

Cvfem_Dispersed_Momentum_Z

Constant

Syntax

Bc Lumped_Flux For Cvfem_Dispersed_Momentum_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | Value = *value*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
value	real	–

Exponential

Syntax

Bc Lumped_Flux For Cvfem_Dispersed_Momentum_Z ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Lumped_Flux For Cvfem_Dispersed_Momentum_Z ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Bc Lumped_Flux For Cvfem_Dispersed_Momentum_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc Lumped_Flux For Cvfem_Dispersed_Momentum_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Lumped_Flux For Cvfem_Dispersed_Momentum_Z ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f	"string"	–

User_Field

Syntax

Bc Lumped_Flux For Cvfem_Dispersed_Momentum_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc Lumped_Flux For Cvfem_Dispersed_Momentum_Z ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc Lumped_Flux For Cvfem_Dispersed_Momentum_Z ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

Cvfem_Div_Projection

Constant

Syntax

Bc Lumped_Flux For Cvfem_Div_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | [Valuexx = *valuexx*] | [Valuexy = *valuexy*] | [Valuexz = *valuexz*] | [Valueyx = *valueyx*] | [Valueyy = *valueyy*] | [Valueyz = *valueyz*] | [Valuezx = *valuezx*] | [Valuezy = *valuezy*] | [Valuezz = *valuezz*] | [Valuet11 = *valuet11*] | [Valuet12 = *valuet12*] | [Valuet13 = *valuet13*] | [Valuet21 = *valuet21*] | [Valuet22 = *valuet22*] | [Valuet23 = *valuet23*] | [Valuet31 = *valuet31*] | [Valuet32 = *valuet32*] | [Valuet33 = *valuet33*])

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
valuexx	real	0
valuexy	real	0
valuexz	real	0
valueyx	real	0
valueyy	real	0
valueyz	real	0
valuezx	real	0
valuezy	real	0
valuezz	real	0
valuet11	real	0
valuet12	real	0
valuet13	real	0
valuet21	real	0
valuet22	real	0
valuet23	real	0
valuet31	real	0
valuet32	real	0
valuet33	real	0

Simple_Interp_Tensor

Syntax

Bc Lumped_Flux For Cvfem_Div_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Simple_Interp_Tensor ([Toggle = *toggle*] | Source = *source*)

Summary

CVFEM divergence projection simple interpolation tensor flux BC

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
source	"string"	–

Simple_Interp_Vector

Syntax

Bc Lumped_Flux For Cvfem_Div_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Simple_Interp_Vector ([Toggle = *toggle*] | Source = *source*)

Summary

CVFEM divergence projection simple interpolation vector flux BC

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
source	"string"	–

Tensor_String_Function

Syntax

Bc Lumped_Flux For Cvfem_Div_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Tensor_String_Function ([Toggle = *toggle*] | F_Xx = *f_xx* | F_Yy = *f_yy* | [F_Xy = *f_xy*] | [F_Zz = *f_zz*] | [F_Xz = *f_xz*] | [F_Yz = *f_yz*] | [F_Yx = *f_yx*] | [F_Zx = *f_zx*] | [F_Zy = *f_zy*])

Summary

Value from a string function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f_xx	"string"	–
f_yy	"string"	–
f_xy	"string"	0.0
f_zz	"string"	–
f_xz	"string"	0.0
f_yz	"string"	0.0
f_yx	"string"	0.0
f_zx	"string"	0.0
f_zy	"string"	0.0

Tensor_String_Function_Symmetric

Syntax

Bc Lumped_Flux For Cvfem_Div_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Tensor_String_Function_Symmetric ([Toggle = *toggle*] | F_Xx = *f_xx* | F_Yy = *f_yy* | [F_Xy = *f_xy*] | [F_Zz = *f_zz*] | [F_Xz = *f_xz*] | [F_Yz = *f_yz*])

Summary

Value from a string function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f_xx	"string"	–
f_yy	"string"	–
f_xy	"string"	0.0
f_zz	"string"	–
f_xz	"string"	0.0
f_yz	"string"	0.0

Cvfem_Energy

Constant

Syntax

Bc Lumped_Flux For Cvfem_Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Data = *data*] | *{flux | value}* = *flux*)

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
data	"string"	–
flux	real	–

Exponential

Syntax

Bc Lumped_Flux For Cvfem_Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Data = *data*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
data	“string”	–
variable	“string”	–
constant	real	0
multiplier	real	1
exponent	real	–

Exponential_Vapor_Cooling

Syntax

Bc Lumped_Flux For Cvfem_Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential_Vapor_Cooling ([Data = *data*] | Tboil = *tboil* | L = *l* | Pa = *pa* | Mw = *mw* | Alpha = *alpha* | [Tl = *tl*] | [Th = *th*])

Summary

Exponential vapor cooling energy flux

Description

Flux is given by:

$$F = P_a L\alpha \exp\left(\frac{L}{R}\left(\frac{1}{T_{boil}} - \frac{1}{T}\right)\right)\sqrt{\frac{1}{2\pi RT}}$$

where R is the specific gas constant ($\bar{R}$/MW). For T less than T_L, zero flux is applied. For T between T_L and T_H, a 3rd order smoothing polynomial is used to provide continuity of values and derivatives between 0 and the expression given above. For T greater than T_H, the above expression is used. If not provided, T_L and T_H are set to 10% below and above T_{Boil}, respectively.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
data	“string”	–
tboil	real	–
l	real	–
pa	real	–
mw	real	–
alpha	real	–
tl	real	-1
th	real	-1

Generalized_Nat_Conv

Syntax

Bc Lumped_Flux For Cvfem_Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Generalized_Nat_Conv ([Data = *data*] | [Multiplier = *multiplier*] | [H = *h*] | [T_Ref = *t_ref*])

Summary

CVFEM energy generalized natural convection flux BC

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
data	"string"	–
multiplier	real	1
h	"string"	–
t_ref	"string"	–

Generalized_Rad

Syntax

Bc Lumped_Flux For Cvfem_Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Generalized_Rad ([Data = *data*] | [Multiplier = *multiplier*] | [T_Ref = *t_ref*])

Summary

Radiative energy flux, with values for emissivity, form factor and irradiation taken from expressions

Description

Flux is:

$$M\epsilon f\left(\sigma T^4 - I\right)$$

M is MULTIPLIER. σ is the Stefan-Boltzmann constant. ϵ and F are expressions for emissivity and form factor. T is surface temperature. I is the expression for irradiation. If I does not already exist, it will be constructed as σT_{ref}^4, where T_{ref} is either provided by the BC Rad Reference Temperature expression or the inline T_{ref} parameter (constant or string function).

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
data	"string"	–
multiplier	real	1
t_ref	"string"	–

Global

Syntax

Bc Lumped_Flux For Cvfem_Energy ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Data = *data*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
data	"string"	–
global_name	"string"	–
component	integer	0

Interface

Syntax

Bc Lumped_Flux For Cvfem_Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Interface [Data = *data*]

Summary

INTERFACE CVFEM energy flux

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
data	“string”	–

Mass_Flux

Syntax

Bc Lumped_Flux For Cvfem_Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Mass_Flux ([Data = *data*] | Inflow_Phi = *Inflow_Phi* | Total_Mdot = *total_mdot*)

Summary

CVFEM energy mass flux BC

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
data	"string"	–
Inflow_Phi	real	–
total_mdot	real	–

Mass_Flux_From_Temperature

Syntax

Bc Lumped_Flux For Cvfem_Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Mass_Flux_From_Temperature ([Data = *data*] | [Temperature = *temperature*] | [Pressure = *pressure*] | [Mass_Fraction_0 = *mass_fraction_0*] | [Mass_Fraction_1 = *mass_fraction_1*] | [Mass_Fraction_2 = *mass_fraction_2*] | [Mass_Fraction_3 = *mass_fraction_3*] | [Mass_Fraction_4 = *mass_fraction_4*] | [Mass_Fraction_5 = *mass_fraction_5*] | [Mass_Fraction_6 = *mass_fraction_6*] | [Mass_Fraction_7 = *mass_fraction_7*] | [Mass_Fraction_8 = *mass_fraction_8*] | Total_Mdot = *total_mdot*)

Summary

CVFEM energy mass flux from temperature flux BC

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
data	"string"	–
temperature	real	0
pressure	real	0
mass_fraction_0	real	0
mass_fraction_1	real	0
mass_fraction_2	real	0
mass_fraction_3	real	0
mass_fraction_4	real	0
mass_fraction_5	real	0
mass_fraction_6	real	0
mass_fraction_7	real	0
mass_fraction_8	real	0
total_mdot	real	–

Nat_Conv

Syntax

Bc Lumped_Flux For Cvfem_Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nat_Conv ([Data = *data*] | H = *h* | T_Ref = *t_ref*)

Summary

CVFEM convective energy flux defined with constant models for heat transfer coefficient and reference temperature

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
data	"string"	–
h	real	–
t_ref	real	–

Nd_Table

Syntax

Bc Lumped_Flux For Cvfem_Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Data = *data*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
data	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Open_Adv_Flow

Syntax

Bc Lumped_Flux For Cvfem_Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Open_Adv_Flow ([Data = *data*] | Far_Field_Entrainment_Value = *Far_Field_Entrainment_Value* | [Pressure = *pressure*] | [Total_Pressure = *total_pressure*])

Summary

CVFEM energy open advective flow flux

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
data	"string"	–
Far_Field_Entrainment_Value	real	–
pressure	real	0
total_pressure	real	0

Open_Adv_Flow_From_Temperature

Syntax

Bc Lumped_Flux For Cvfem_Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Open_Adv_Flow_From_Temperature ([Data = *data*] | [Temperature = *temperature*] | [Pressure = *pressure*] | [Mass_Fraction_0 = *mass_fraction_0*] | [Mass_Fraction_1 = *mass_fraction_1*] | [Mass_Fraction_2 = *mass_fraction_2*] | [Mass_Fraction_3 = *mass_fraction_3*] | [Mass_Fraction_4 = *mass_fraction_4*] | [Mass_Fraction_5 = *mass_fraction_5*] | [Mass_Fraction_6 = *mass_fraction_6*] | [Mass_Fraction_7 = *mass_fraction_7*] | [Mass_Fraction_8 = *mass_fraction_8*] | [Total_Pressure = *total_pressure*])

Summary

CVFEM energy open advective flow from temperature flux

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
data	“string”	–
temperature	real	0
pressure	real	0
mass_fraction_0	real	0
mass_fraction_1	real	0
mass_fraction_2	real	0
mass_fraction_3	real	0
mass_fraction_4	real	0
mass_fraction_5	real	0
mass_fraction_6	real	0
mass_fraction_7	real	0
mass_fraction_8	real	0
total_pressure	real	0

Polynomial

Syntax

Bc Lumped_Flux For Cvfem_Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Data = *data*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
data	"string"	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Porous_Robin_Coupled

Syntax

Bc Lumped_Flux For Cvfem_Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Porous_Robin_Coupled ([Data = *data*] | Enthalpy_Field = *enthalpy_field* | Enthalpy_Diffusive_Flux_Field = *enthalpy_diffusive_flux_field* | Averaged_Coeff_Field = *averaged_coeff_field*)

Summary

Boundary condition for loose coupling of enthalpy to a porous region

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
data	"string"	–
enthalpy_field	"string"	–
enthalpy_diffusive_flux_field	"string"	–
averaged_coeff_field	"string"	–

Porous_Robin_Coupled_One_Region

Syntax

Bc Lumped_Flux For Cvfem_Energy ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Porous_Robin_Coupled_One_Region [Data = *data*]

Summary

Boundary condition for loose coupling of enthalpy to a porous region

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
data	"string"	–

Porous_Robin_Coupled_With_Solid_Phase_Convection

Syntax

Bc Lumped_Flux For Cvfem_Energy ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Porous_Robin_Coupled_With_Solid_Phase_Convection ([Data = *data*] | [Multiplier = *multiplier*] | Enthalpy_Field = *enthalpy_field* | Enthalpy_Diffusive_Flux_Field = *enthalpy_diffusive_flux_field* | Solid_Phase_Temperature_Field = *solid_phase_temperature_field* | Porosity_Field = *porosity_field* | Averaged_Coeff_Field = *averaged_coeff_field* | Volumetric_Heat_Transfer_Coeff_Field = *volumetric_heat_transfer_coeff_field* | Specific_Surface_Area_Field = *specific_surface_area_field*)

Summary

Boundary condition for loose coupling of enthalpy to a porous region

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
data	“string”	–
multiplier	real	1
enthalpy_field	“string”	–
enthalpy_diffusive_flux_field	“string”	–
solid_phase_temperature_field	“string”	–
porosity_field	“string”	–
averaged_coeff_field	“string”	–
volumetric_heat_transfer_coeff_field	“string”	–
specific_surface_area_field	“string”	–

Porous_Robin_Coupled_With_Solid_Phase_Convection_One_Region

Syntax

Bc Lumped_Flux For Cvfem_Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Porous_Robin_Coupled_With_Solid_Phase_Convection_One_Region ([Data = *data*] | [Multiplier = *multiplier*])

Summary

Boundary condition for coupling of enthalpy to a porous region in the same region.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
data	“string”	–
multiplier	real	1

Rad

Syntax

Bc Lumped_Flux For Cvfem_Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Rad ([Data = *data*] | Crad = *crad* | T_Ref = *t_ref*)

Summary

Radiative energy flux defined with constant models for emissivity and reference temperature.

Description

The applied flux is given by:

$$c_{rad}\left(T^4 - T_{ref}^4\right)$$

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
data	"string"	–
crad	real	–
t_ref	real	–

Ray_Trace

Syntax

Bc Lumped_Flux For Cvfem_Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Ray_Trace ([Data = *data*] | Ray_Source = *ray_source* | [Emission = *emission*])

Summary

Apply a ray tracing energy flux

Description

Adds an energy flux based on the intersection of rays defined in a ray tracing source block for the equation system with name given by the RAY_SOURCE parameter with the surface. The optional EMISSION parameter also imposes a σT^4 blackbody radiative emission term

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
data	"string"	–
ray_source	"string"	–
emission	"string"	FALSE

Scalar_String_Function

Syntax

Bc Lumped_Flux For Cvfem_Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Data = *data*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
data	“string”	–
f	“string”	–

User_Field

Syntax

Bc Lumped_Flux For Cvfem_Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Data = *data*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
data	"string"	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc Lumped_Flux For Cvfem_Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Data = *data*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
data	“string”	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Bc Lumped_Flux For Cvfem_Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Data = *data*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
data	“string”	–
Name	“string”	–
plugin_parameters	“string”	–

Wall_Function_From_Cht_Temperature

Syntax

Bc Lumped_Flux For Cvfem_Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Wall_Function_From_Cht_Temperature ([Data = *data*] | Wall_Friction_Factor = *wall_friction_factor*)

Summary

CVFEM energy wall function from conjugate heat transfer temperature flux BC

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
data	“string”	–
wall_friction_factor	real	–

Wall_Function_From_Temperature

Syntax

Bc Lumped_Flux For Cvfem_Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Wall_Function_From_Temperature ([Data = *data*] | *{t | value}* = *t* | Wall_Friction_Factor = *wall_friction_factor*)

Summary

CVFEM energy wall function from temperature flux BC

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
data	"string"	–
t	real	–
wall_friction_factor	real	–

Cvfem_Level_Set

Open_Adv_Flow

Syntax

Bc Lumped_Flux For Cvfem_Level_Set ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Open_Adv_Flow ([Toggle = *toggle*] | [Pressure = *pressure*] | [Total_Pressure = *total_pressure*] | Far_Field_Entrainment_Value = *far_field_entrainment_value*)

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
pressure	real	0
total_pressure	real	0
far_field_entrainment_value	real	–

Open_Nc_Adv_Flow

Syntax

Bc Lumped_Flux For Cvfem_Level_Set ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Open_Nc_Adv_Flow ([Toggle = *toggle*] | Far_Field_Entrainment_Value = *far_field_entrainment_value*)

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
far_field_entrainment_value	real	–

Cvfem_Lumped_Div_Projection

Constant

Syntax

Bc Lumped_Flux For Cvfem_Lumped_Div_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | [Valuexx = *valuexx*] | [Valuexy = *valuexy*] | [Valuexz = *valuexz*] | [Valueyx = *valueyx*] | [Valueyy = *valueyy*] | [Valueyz = *valueyz*] | [Valuezx = *valuezx*] | [Valuezy = *valuezy*] | [Valuezz = *valuezz*] | [Valuet11 = *valuet11*] | [Valuet12 = *valuet12*] | [Valuet13 = *valuet13*] | [Valuet21 = *valuet21*] | [Valuet22 = *valuet22*] | [Valuet23 = *valuet23*] | [Valuet31 = *valuet31*] | [Valuet32 = *valuet32*] | [Valuet33 = *valuet33*])

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
valuexx	real	0
valuexy	real	0
valuexz	real	0
valueyx	real	0
valueyy	real	0
valueyz	real	0
valuezx	real	0
valuezy	real	0
valuezz	real	0
valuet11	real	0
valuet12	real	0
valuet13	real	0
valuet21	real	0
valuet22	real	0
valuet23	real	0
valuet31	real	0
valuet32	real	0
valuet33	real	0

Simple_Interp_Tensor

Syntax

Bc Lumped_Flux For Cvfem_Lumped_Div_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Simple_Interp_Tensor ([Toggle = *toggle*] | Source = *source*)

Summary

CVFEM lumped divergence projection simple interpolation tensor flux BC

Description

The following options are present in most boundary conditions/sources. The

meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
source	"string"	–

Simple_Interp_Vector

Syntax

Bc Lumped_Flux For Cvfem_Lumped_Div_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Simple_Interp_Vector ([Toggle = *toggle*] | Source = *source*)

Summary

CVFEM lumped divergence projection simple interpolation vector flux BC

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
source	"string"	–

Tensor_String_Function

Syntax

Bc Lumped_Flux For Cvfem_Lumped_Div_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Tensor_String_Function ([Toggle = *toggle*] | F_Xx = *f_xx* | F_Yy = *f_yy* | [F_Xy = *f_xy*] | [F_Zz = *f_zz*] | [F_Xz = *f_xz*] | [F_Yz = *f_yz*] | [F_Yx = *f_yx*] | [F_Zx = *f_zx*] | [F_Zy = *f_zy*])

Summary

Value from a string function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
f_xx	“string”	–
f_yy	“string”	–
f_xy	“string”	0.0
f_zz	“string”	–
f_xz	“string”	0.0
f_yz	“string”	0.0
f_yx	“string”	0.0
f_zx	“string”	0.0
f_zy	“string”	0.0

Tensor_String_Function_Symmetric

Syntax

Bc Lumped_Flux For Cvfem_Lumped_Div_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Tensor_String_Function_Symmetric ([Toggle = *toggle*] | F_Xx = *f_xx* | F_Yy = *f_yy* | [F_Xy = *f_xy*] | [F_Zz = *f_zz*] | [F_Xz = *f_xz*] | [F_Yz = *f_yz*])

Summary

Value from a string function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f_xx	"string"	–
f_yy	"string"	–
f_xy	"string"	0.0
f_zz	"string"	–
f_xz	"string"	0.0
f_yz	"string"	0.0

Cvfem_Lumped_Projection

Constant

Syntax

Bc Lumped_Flux For Cvfem_Lumped_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | Value = *value*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
value	real	–

Exponential

Syntax

Bc Lumped_Flux For Cvfem_Lumped_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Lumped_Flux For Cvfem_Lumped_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Bc Lumped_Flux For Cvfem_Lumped_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Open

Syntax

Bc Lumped_Flux For Cvfem_Lumped_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Open ([Toggle = *toggle*] | [Total_Pressure = *total_pressure*] | [Pressure = *pressure*])

Summary

CVFEM lumped projection open flux BC

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
total_pressure	real	0
pressure	real	0

Polynomial

Syntax

Bc Lumped_Flux For Cvfem_Lumped_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Lumped_Flux For Cvfem_Lumped_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f	"string"	–

Simple_Interp

Syntax

Bc Lumped_Flux For Cvfem_Lumped_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Simple_Interp [Toggle = *toggle*]

Summary

CVFEM lumped projection simple interpolation flux BC

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–

User_Field

Syntax

Bc Lumped_Flux For Cvfem_Lumped_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
scaling	real	1
global_var	“string”	–
time_function	“string”	–

User_Function

Syntax

Bc Lumped_Flux For Cvfem_Lumped_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc Lumped_Flux For Cvfem_Lumped_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

Cvfem_Mass_Fraction

Constant

Syntax

Bc Lumped_Flux For Cvfem_Mass_Fraction ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | Value = *value*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
value	real	–

Exponential

Syntax

Bc Lumped_Flux For Cvfem_Mass_Fraction ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Lumped_Flux For Cvfem_Mass_Fraction ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
global_name	"string"	–
component	integer	0

Mass_Flux

Syntax

Bc Lumped_Flux For Cvfem_Mass_Fraction ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Mass_Flux ([Toggle = *toggle*] | Total_Mdot = *total_mdot* | [Inflow_Phi = *inflow_phi*])

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
total_mdot	real	–
inflow_phi	real	0

Nd_Table

Syntax

Bc Lumped_Flux For Cvfem_Mass_Fraction ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Open_Adv_Flow

Syntax

Bc Lumped_Flux For Cvfem_Mass_Fraction ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Open_Adv_Flow ([Toggle = *toggle*] | [Pressure = *pressure*] | [Total_Pressure = *total_pressure*] | [Far_Field_Entrainment_Value = *far_field_entrainment_value*])

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
pressure	real	0
total_pressure	real	0
far_field_entrainment_value	real	0

Polynomial

Syntax

Bc Lumped_Flux For Cvfem_Mass_Fraction ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Porous_Robin_Coupled

Syntax

Bc Lumped_Flux For Cvfem_Mass_Fraction ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Porous_Robin_Coupled ([Toggle = *toggle*] | Mass_Fraction_Field = *mass_fraction_field* | Mass_Fraction_Diffusive_Flux_Field = *mass_fraction_diffusive_flux_field* | Averaged_Coeff_Field = *averaged_coeff_field*)

Summary

Boundary condition for loose coupling of scalars to a free fluid region

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
mass_fraction_field	“string”	–
mass_fraction_diffusive_flux_field	“string”	–
averaged_coeff_field	“string”	–

Porous_Robin_Coupled_One_Region

Syntax

Bc Lumped_Flux For Cvfem_Mass_Fraction ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Porous_Robin_Coupled_One_Region [Toggle = *toggle*]

Summary

Boundary condition for loose coupling of scalars to a free fluid region

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–

Scalar_String_Function

Syntax

Bc Lumped_Flux For Cvfem_Mass_Fraction ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f	"string"	–

User_Field

Syntax

Bc Lumped_Flux For Cvfem_Mass_Fraction ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc Lumped_Flux For Cvfem_Mass_Fraction ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc Lumped_Flux For Cvfem_Mass_Fraction ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

Cvfem_Mixture_Fraction

Constant

Syntax

Bc Lumped_Flux For Cvfem_Mixture_Fraction ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | Value = *value*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
value	real	–

Exponential

Syntax

Bc Lumped_Flux For Cvfem_Mixture_Fraction ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Free_Open_Flow

Syntax

Bc Lumped_Flux For Cvfem_Mixture_Fraction ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Free_Open_Flow [Toggle = *toggle*]

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–

Global

Syntax

Bc Lumped_Flux For Cvfem_Mixture_Fraction ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
global_name	“string”	–
component	integer	0

Mass_Flux

Syntax

Bc Lumped_Flux For Cvfem_Mixture_Fraction ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Mass_Flux ([Toggle = *toggle*] | Total_Mdot = *total_mdot* | [Inflow_Phi = *inflow_phi*])

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
total_mdot	real	–
inflow_phi	real	0

Nd_Table

Syntax

Bc Lumped_Flux For Cvfem_Mixture_Fraction ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Open_Adv_Flow

Syntax

Bc Lumped_Flux For Cvfem_Mixture_Fraction ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Open_Adv_Flow ([Toggle = *toggle*] | [Pressure = *pressure*] | [Total_Pressure = *total_pressure*] | [Far_Field_Entrainment_Value = *far_field_entrainment_value*])

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
pressure	real	0
total_pressure	real	0
far_field_entrainment_value	real	0

Polynomial

Syntax

Bc Lumped_Flux For Cvfem_Mixture_Fraction ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Lumped_Flux For Cvfem_Mixture_Fraction ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f	"string"	–

User_Field

Syntax

Bc Lumped_Flux For Cvfem_Mixture_Fraction ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc Lumped_Flux For Cvfem_Mixture_Fraction ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc Lumped_Flux For Cvfem_Mixture_Fraction ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

Cvfem_Momentum

Constant

Syntax

Bc Lumped_Flux For Cvfem_Momentum ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | [Valuex = *valuex*] | [Valuey = *valuey*] | [Valuez = *valuez*])

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
valuex	real	0
valuey	real	0
valuez	real	0

Constant_Traction

Syntax

Bc Lumped_Flux For Cvfem_Momentum ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant_Traction ([Toggle = *toggle*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
x	real	0
y	real	0
z	real	0

Exponential_Vapor_Recoil_Pressure

Syntax

Bc Lumped_Flux For Cvfem_Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential_Vapor_Recoil_Pressure ([Toggle = *toggle*] | Tboil = *tboil* | L = *l* | Pa = *pa* | [Beta = *beta*] | Mw = *mw* | [Tl = *tl*] | [Th = *th*])

Summary

Exponential vapor recoil pressure momentum flux

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
tboil	real	–
l	real	–
pa	real	–
beta	real	1
mw	real	–
tl	real	-1
th	real	-1

Free_Open_Flow

Syntax

Bc Lumped_Flux For Cvfem_Momentum ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Free_Open_Flow ([Toggle = *toggle*] | Pressure = *pressure*)

Summary

Open flow for momentum; apply the full stress

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
pressure	real	–

Interface

Syntax

Bc Lumped_Flux For Cvfem_Momentum ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Interface [Toggle = *toggle*]

Summary

INTERFACE CVFEM momentum flux

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–

Mass_Flux

Syntax

Bc Lumped_Flux For Cvfem_Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Mass_Flux ([Toggle = *toggle*] | Total_Mdot = *total_mdot* | [Temperature = *temperature*] | [Pressure = *pressure*] | [Mass_Fraction_0 = *mass_fraction_0*] | [Mass_Fraction_1 = *mass_fraction_1*] | [Mass_Fraction_2 = *mass_fraction_2*] | [Mass_Fraction_3 = *mass_fraction_3*] | [Mass_Fraction_4 = *mass_fraction_4*] | [Mass_Fraction_5 = *mass_fraction_5*] | [Mass_Fraction_6 = *mass_fraction_6*] | [Mass_Fraction_7 = *mass_fraction_7*] | [Mass_Fraction_8 = *mass_fraction_8*])

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
total_mdot	real	–
temperature	real	0
pressure	real	0
mass_fraction_0	real	0
mass_fraction_1	real	0
mass_fraction_2	real	0
mass_fraction_3	real	0
mass_fraction_4	real	0
mass_fraction_5	real	0
mass_fraction_6	real	0
mass_fraction_7	real	0
mass_fraction_8	real	0

Open_Adv_Flow

Syntax

Bc Lumped_Flux For Cvfem_Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Open_Adv_Flow ([Toggle = *toggle*] | [Pressure = *pressure*] | [Total_Pressure = *total_pressure*] | [Coflow_Entrainment_Value_X = *coflow_entrainment_value_x*] | [Coflow_Entrainment_Value_Y = *coflow_entrainment_value_y*] | [Coflow_Entrainment_Value_Z = *coflow_entrainment_value_z*])

Summary

CVFEM momentum open advective flow flux BC

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is

active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
pressure	real	0
total_pressure	real	0
coflow_entrainment_value_x	real	0
coflow_entrainment_value_y	real	0
coflow_entrainment_value_z	real	0

Open_Flow

Syntax

Bc Lumped_Flux For Cvfem_Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Open_Flow ([Toggle = *toggle*] | [Pressure = *pressure*] | [Total_Pressure = *total_pressure*])

Summary

Open flow for momentum; enforce normal gradient to be zero

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
pressure	real	0
total_pressure	real	0

Slip

Syntax

Bc Lumped_Flux For Cvfem_Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Slip ([Toggle = *toggle*] | [Vs_X = *vs_x*] | [Vs_Y = *vs_y*] | [Vs_Z = *vs_z*] | Beta = *beta*)

Summary

Slip momentum flux

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
vs_x	real	0
vs_y	real	0
vs_z	real	0
beta	real	–

Slip_Length

Syntax

Bc Lumped_Flux For Cvfem_Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Slip_Length ([Toggle = *toggle*] | [Vs_X = *vs_x*] | [Vs_Y = *vs_y*] | [Vs_Z = *vs_z*] | Beta = *beta*)

Summary

Slip length momentum flux

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
vs_x	real	0
vs_y	real	0
vs_z	real	0
beta	real	–

Symmetry_Flow

Syntax

Bc Lumped_Flux For Cvfem_Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Symmetry_Flow [Toggle = *toggle*]

Summary

CVFEM momentum symmetry flow flux BC

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–

Vector_String_Function

Syntax

Bc Lumped_Flux For Cvfem_Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Vector_String_Function ([Toggle = *toggle*] | F_X = *f_x* | F_Y = *f_y* | [F_Z = *f_z*])

Summary

Value from a string function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f_x	"string"	–
f_y	"string"	–
f_z	"string"	–

Wall_Function

Syntax

Bc Lumped_Flux For Cvfem_Momentum ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Wall_Function ([Toggle = *toggle*] | [Wall_Velocity_X = *wall_velocity_x*] | [Wall_Velocity_Y = *wall_velocity_y*] | [Wall_Velocity_Z = *wall_velocity_z*] | Wall_Friction_Factor = *wall_friction_factor*)

Summary

CVFEM momentum wall function flux BC

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
wall_velocity_x	real	0
wall_velocity_y	real	0
wall_velocity_z	real	0
wall_friction_factor	real	–

Cvfem_Momentum_X

Constant

Syntax

Bc Lumped_Flux For Cvfem_Momentum_X ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent*

Name ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | Value = *value*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
value	real	–

Exponential

Syntax

Bc Lumped_Flux For Cvfem_Momentum_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Lumped_Flux For Cvfem_Momentum_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Bc Lumped_Flux For Cvfem_Momentum_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc Lumped_Flux For Cvfem_Momentum_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Lumped_Flux For Cvfem_Momentum_X ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f	"string"	–

User_Field

Syntax

Bc Lumped_Flux For Cvfem_Momentum_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc Lumped_Flux For Cvfem_Momentum_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc Lumped_Flux For Cvfem_Momentum_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

Cvfem_Momentum_Y

Constant

Syntax

Bc Lumped_Flux For Cvfem_Momentum_Y ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | Value = *value*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
value	real	–

Exponential

Syntax

Bc Lumped_Flux For Cvfem_Momentum_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Lumped_Flux For Cvfem_Momentum_Y ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Bc Lumped_Flux For Cvfem_Momentum_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc Lumped_Flux For Cvfem_Momentum_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Lumped_Flux For Cvfem_Momentum_Y ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f	"string"	–

User_Field

Syntax

Bc Lumped_Flux For Cvfem_Momentum_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc Lumped_Flux For Cvfem_Momentum_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc Lumped_Flux For Cvfem_Momentum_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

Cvfem_Momentum_Z

Constant

Syntax

Bc Lumped_Flux For Cvfem_Momentum_Z ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | Value = *value*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
value	real	–

Exponential

Syntax

Bc Lumped_Flux For Cvfem_Momentum_Z ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Lumped_Flux For Cvfem_Momentum_Z ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Bc Lumped_Flux For Cvfem_Momentum_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc Lumped_Flux For Cvfem_Momentum_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Lumped_Flux For Cvfem_Momentum_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f	"string"	–

User_Field

Syntax

Bc Lumped_Flux For Cvfem_Momentum_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc Lumped_Flux For Cvfem_Momentum_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc Lumped_Flux For Cvfem_Momentum_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

Cvfem_Pressure_Edge_Gradient_Projection

Constant

Syntax

Bc Lumped_Flux For Cvfem_Pressure_Edge_Gradient_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | Value = *value*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
value	real	–

Exponential

Syntax

Bc Lumped_Flux For Cvfem_Pressure_Edge_Gradient_Projection ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Lumped_Flux For Cvfem_Pressure_Edge_Gradient_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Bc Lumped_Flux For Cvfem_Pressure_Edge_Gradient_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc Lumped_Flux For Cvfem_Pressure_Edge_Gradient_Projection ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Lumped_Flux For Cvfem_Pressure_Edge_Gradient_Projection ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f	"string"	–

Simple_Interp

Syntax

Bc Lumped_Flux For Cvfem_Pressure_Edge_Gradient_Projection ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Simple_Interp [Toggle = *toggle*]

Summary

CVFEM edge-based projection simple interpolation flux BC

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–

User_Field

Syntax

Bc Lumped_Flux For Cvfem_Pressure_Edge_Gradient_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
scaling	real	1
global_var	“string”	–
time_function	“string”	–

User_Function

Syntax

Bc Lumped_Flux For Cvfem_Pressure_Edge_Gradient_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc Lumped_Flux For Cvfem_Pressure_Edge_Gradient_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

Cvfem_Projection

Constant

Syntax

Bc Lumped_Flux For Cvfem_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | Value = *value*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
value	real	–

Exponential

Syntax

Bc Lumped_Flux For Cvfem_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Lumped_Flux For Cvfem_Projection ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Bc Lumped_Flux For Cvfem_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Open

Syntax

Bc Lumped_Flux For Cvfem_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Open ([Toggle = *toggle*] | [Total_Pressure = *total_pressure*] | [Pressure = *pressure*])

Summary

CVFEM projection open flux BC

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
total_pressure	real	0
pressure	real	0

Polynomial

Syntax

Bc Lumped_Flux For Cvfem_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Lumped_Flux For Cvfem_Projection ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f	"string"	–

Simple_Interp

Syntax

Bc Lumped_Flux For Cvfem_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Simple_Interp [Toggle = *toggle*]

Summary

CVFEM projection simple interpolation flux BC

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–

User_Field

Syntax

Bc Lumped_Flux For Cvfem_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
scaling	real	1
global_var	“string”	–
time_function	“string”	–

User_Function

Syntax

Bc Lumped_Flux For Cvfem_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc Lumped_Flux For Cvfem_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

Cvfem_Sdr_Edge_Gradient_Projection

Constant

Syntax

Bc Lumped_Flux For Cvfem_Sdr_Edge_Gradient_Projection ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | Value = *value*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
value	real	–

Exponential

Syntax

Bc Lumped_Flux For Cvfem_Sdr_Edge_Gradient_Projection ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Lumped_Flux For Cvfem_Sdr_Edge_Gradient_Projection ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Bc Lumped_Flux For Cvfem_Sdr_Edge_Gradient_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc Lumped_Flux For Cvfem_Sdr_Edge_Gradient_Projection ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Lumped_Flux For Cvfem_Sdr_Edge_Gradient_Projection ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f	"string"	–

Simple_Interp

Syntax

Bc Lumped_Flux For Cvfem_Sdr_Edge_Gradient_Projection ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Simple_Interp [Toggle = *toggle*]

Summary

CVFEM edge-based projection simple interpolation flux BC

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–

User_Field

Syntax

Bc Lumped_Flux For Cvfem_Sdr_Edge_Gradient_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
scaling	real	1
global_var	“string”	–
time_function	“string”	–

User_Function

Syntax

Bc Lumped_Flux For Cvfem_Sdr_Edge_Gradient_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc Lumped_Flux For Cvfem_Sdr_Edge_Gradient_Projection ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

Cvfem_Solvent_Continuity

Constant

Syntax

Bc Lumped_Flux For Cvfem_Solvent_Continuity ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | Value = *value*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
value	real	–

Exponential

Syntax

Bc Lumped_Flux For Cvfem_Solvent_Continuity ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Lumped_Flux For Cvfem_Solvent_Continuity ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
global_name	"string"	–
component	integer	0

Inflow

Syntax

Bc Lumped_Flux For Cvfem_Solvent_Continuity ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Inflow [Toggle = *toggle*]

Summary

Inflow mass flux for continuity equation; nodal interpolation

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–

Nd_Table

Syntax

Bc Lumped_Flux For Cvfem_Solvent_Continuity ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Open_Flow

Syntax

Bc Lumped_Flux For Cvfem_Solvent_Continuity ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Open_Flow ([Toggle = *toggle*] | [Pressure = *pressure*] | [Total_Pressure = *total_pressure*])

Summary

Open bc for continuity

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
pressure	real	0
total_pressure	real	0

Polynomial

Syntax

Bc Lumped_Flux For Cvfem_Solvent_Continuity ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
variable	“string”	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Lumped_Flux For Cvfem_Solvent_Continuity ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f	"string"	–

Simple_Inflow

Syntax

Bc Lumped_Flux For Cvfem_Solvent_Continuity ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Simple_Inflow ([Toggle = *toggle*] | *{density | value}* = *Density* | [Velocity_X = *Velocity_X*] | [Velocity_Y = *Velocity_Y*] | [Velocity_Z = *Velocity_Z*])

Summary

Inflow mass flux for continuity equation; specified density and velocity

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Density	real	–
Velocity_X	real	0
Velocity_Y	real	0
Velocity_Z	real	0

User_Field

Syntax

Bc Lumped_Flux For Cvfem_Solvent_Continuity ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc Lumped_Flux For Cvfem_Solvent_Continuity ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc Lumped_Flux For Cvfem_Solvent_Continuity ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

Cvfem_Solvent_Momentum

Constant

Syntax

Bc Lumped_Flux For Cvfem_Solvent_Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | [Valuex = *valuex*] | [Valuey = *valuey*] | [Valuez = *valuez*])

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
valuex	real	0
valuey	real	0
valuez	real	0

Open_Adv_Flow

Syntax

Bc Lumped_Flux For Cvfem_Solvent_Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Open_Adv_Flow ([Toggle = *toggle*] | [Pressure = *pressure*] | [Total_Pressure = *total_pressure*] | [Coflow_Entrainment_Value_X = *coflow_entrainment_value_x*] | [Coflow_Entrainment_Value_Y = *coflow_entrainment_value_y*] | [Coflow_Entrainment_Value_Z = *coflow_entrainment_value_z*])

Summary

CVFEM solvent momentum open advective flow flux BC

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
pressure	real	0
total_pressure	real	0
coflow_entrainment_value_x	real	0
coflow_entrainment_value_y	real	0
coflow_entrainment_value_z	real	0

Open_Flow

Syntax

Bc Lumped_Flux For Cvfem_Solvent_Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Open_Flow ([Toggle = *toggle*] | [Pressure = *pressure*] | [Total_Pressure = *total_pressure*])

Summary

CVFEM solvent momentum open flow flux BC

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
pressure	real	0
total_pressure	real	0

Symmetry_Flow

Syntax

Bc Lumped_Flux For Cvfem_Solvent_Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Symmetry_Flow [Toggle = *toggle*]

Summary

CVFEM solvent momentum symmetry flow flux BC

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–

Vector_String_Function

Syntax

Bc Lumped_Flux For Cvfem_Solvent_Momentum ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Vector_String_Function ([Toggle = *toggle*] | F_X = *f_x* | F_Y = *f_y* | [F_Z = *f_z*])

Summary

Value from a string function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
f_x	“string”	–
f_y	“string”	–
f_z	“string”	–

Cvfem_Solvent_Momentum_X

Constant

Syntax

Bc Lumped_Flux For Cvfem_Solvent_Momentum_X ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | Value = *value*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
value	real	–

Exponential

Syntax

Bc Lumped_Flux For Cvfem_Solvent_Momentum_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Lumped_Flux For Cvfem_Solvent_Momentum_X ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Bc Lumped_Flux For Cvfem_Solvent_Momentum_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc Lumped_Flux For Cvfem_Solvent_Momentum_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Lumped_Flux For Cvfem_Solvent_Momentum_X ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f	"string"	–

User_Field

Syntax

Bc Lumped_Flux For Cvfem_Solvent_Momentum_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc Lumped_Flux For Cvfem_Solvent_Momentum_X ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc Lumped_Flux For Cvfem_Solvent_Momentum_X ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

Cvfem_Solvent_Momentum_Y

Constant

Syntax

Bc Lumped_Flux For Cvfem_Solvent_Momentum_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | Value = *value*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
value	real	–

Exponential

Syntax

Bc Lumped_Flux For Cvfem_Solvent_Momentum_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Lumped_Flux For Cvfem_Solvent_Momentum_Y ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Bc Lumped_Flux For Cvfem_Solvent_Momentum_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc Lumped_Flux For Cvfem_Solvent_Momentum_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Lumped_Flux For Cvfem_Solvent_Momentum_Y ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f	"string"	–

User_Field

Syntax

Bc Lumped_Flux For Cvfem_Solvent_Momentum_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc Lumped_Flux For Cvfem_Solvent_Momentum_Y ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc Lumped_Flux For Cvfem_Solvent_Momentum_Y ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
Name	“string”	–
plugin_parameters	“string”	–

Cvfem_Solvent_Momentum_Z

Constant

Syntax

Bc Lumped_Flux For Cvfem_Solvent_Momentum_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | Value = *value*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
value	real	–

Exponential

Syntax

Bc Lumped_Flux For Cvfem_Solvent_Momentum_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Lumped_Flux For Cvfem_Solvent_Momentum_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Bc Lumped_Flux For Cvfem_Solvent_Momentum_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc Lumped_Flux For Cvfem_Solvent_Momentum_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Lumped_Flux For Cvfem_Solvent_Momentum_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f	"string"	–

User_Field

Syntax

Bc Lumped_Flux For Cvfem_Solvent_Momentum_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc Lumped_Flux For Cvfem_Solvent_Momentum_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc Lumped_Flux For Cvfem_Solvent_Momentum_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

Cvfem_Specific_Dissipation_Rate

Constant

Syntax

Bc Lumped_Flux For Cvfem_Specific_Dissipation_Rate ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | Value = *value*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
value	real	–

Exponential

Syntax

Bc Lumped_Flux For Cvfem_Specific_Dissipation_Rate ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Lumped_Flux For Cvfem_Specific_Dissipation_Rate ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
global_name	"string"	–
component	integer	0

Mass_Flux

Syntax

Bc Lumped_Flux For Cvfem_Specific_Dissipation_Rate ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Mass_Flux ([Toggle = *toggle*] | Total_Mdot = *total_mdot* | [Inflow_Phi = *inflow_phi*])

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
total_mdot	real	–
inflow_phi	real	0

Nd_Table

Syntax

Bc Lumped_Flux For Cvfem_Specific_Dissipation_Rate ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Open_Adv_Flow

Syntax

Bc Lumped_Flux For Cvfem_Specific_Dissipation_Rate ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Open_Adv_Flow ([Toggle = *toggle*] | [Pressure = *pressure*] | [Total_Pressure = *total_pressure*] | [Far_Field_Entrainment_Value = *far_field_entrainment_value*])

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
pressure	real	0
total_pressure	real	0
far_field_entrainment_value	real	0

Polynomial

Syntax

Bc Lumped_Flux For Cvfem_Specific_Dissipation_Rate ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
variable	“string”	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Lumped_Flux For Cvfem_Specific_Dissipation_Rate ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
f	“string”	–

User_Field

Syntax

Bc Lumped_Flux For Cvfem_Specific_Dissipation_Rate ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc Lumped_Flux For Cvfem_Specific_Dissipation_Rate ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc Lumped_Flux For Cvfem_Specific_Dissipation_Rate ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

Cvfem_Temperature_Edge_Gradient_Projection

Constant

Syntax

Bc Lumped_Flux For Cvfem_Temperature_Edge_Gradient_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | Value = *value*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
value	real	–

Exponential

Syntax

Bc Lumped_Flux For Cvfem_Temperature_Edge_Gradient_Projection ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Lumped_Flux For Cvfem_Temperature_Edge_Gradient_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Bc Lumped_Flux For Cvfem_Temperature_Edge_Gradient_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc Lumped_Flux For Cvfem_Temperature_Edge_Gradient_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Lumped_Flux For Cvfem_Temperature_Edge_Gradient_Projection ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f	"string"	–

Simple_Interp

Syntax

Bc Lumped_Flux For Cvfem_Temperature_Edge_Gradient_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Simple_Interp [Toggle = *toggle*]

Summary

CVFEM edge-based projection simple interpolation flux BC

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–

User_Field

Syntax

Bc Lumped_Flux For Cvfem_Temperature_Edge_Gradient_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
scaling	real	1
global_var	“string”	–
time_function	“string”	–

User_Function

Syntax

Bc Lumped_Flux For Cvfem_Temperature_Edge_Gradient_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc Lumped_Flux For Cvfem_Temperature_Edge_Gradient_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

Cvfem_Tke_Edge_Gradient_Projection

Constant

Syntax

Bc Lumped_Flux For Cvfem_Tke_Edge_Gradient_Projection ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | Value = *value*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
value	real	–

Exponential

Syntax

Bc Lumped_Flux For Cvfem_Tke_Edge_Gradient_Projection ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Lumped_Flux For Cvfem_Tke_Edge_Gradient_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Bc Lumped_Flux For Cvfem_Tke_Edge_Gradient_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc Lumped_Flux For Cvfem_Tke_Edge_Gradient_Projection ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Lumped_Flux For Cvfem_Tke_Edge_Gradient_Projection ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f	"string"	–

Simple_Interp

Syntax

Bc Lumped_Flux For Cvfem_Tke_Edge_Gradient_Projection ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Simple_Interp [Toggle = *toggle*]

Summary

CVFEM edge-based projection simple interpolation flux BC

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–

User_Field

Syntax

Bc Lumped_Flux For Cvfem_Tke_Edge_Gradient_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
scaling	real	1
global_var	“string”	–
time_function	“string”	–

User_Function

Syntax

Bc Lumped_Flux For Cvfem_Tke_Edge_Gradient_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc Lumped_Flux For Cvfem_Tke_Edge_Gradient_Projection ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

Cvfem_Turbulence_Dissipation_Rate

Constant

Syntax

Bc Lumped_Flux For Cvfem_Turbulence_Dissipation_Rate ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | Value = *value*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
value	real	–

Exponential

Syntax

Bc Lumped_Flux For Cvfem_Turbulence_Dissipation_Rate ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Free_Open_Flow

Syntax

Bc Lumped_Flux For Cvfem_Turbulence_Dissipation_Rate ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Free_Open_Flow [Toggle = *toggle*]

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–

Global

Syntax

Bc Lumped_Flux For Cvfem_Turbulence_Dissipation_Rate ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
global_name	“string”	–
component	integer	0

Mass_Flux

Syntax

Bc Lumped_Flux For Cvfem_Turbulence_Dissipation_Rate ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Mass_Flux ([Toggle = *toggle*] | Total_Mdot = *total_mdot* | [Inflow_Phi = *inflow_phi*])

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
total_mdot	real	–
inflow_phi	real	0

Nd_Table

Syntax

Bc Lumped_Flux For Cvfem_Turbulence_Dissipation_Rate ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Open_Adv_Flow

Syntax

Bc Lumped_Flux For Cvfem_Turbulence_Dissipation_Rate ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Open_Adv_Flow ([Toggle = *toggle*] | [Pressure = *pressure*] | [Total_Pressure = *total_pressure*] | [Far_Field_Entrainment_Value = *far_field_entrainment_value*])

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
pressure	real	0
total_pressure	real	0
far_field_entrainment_value	real	0

Polynomial

Syntax

Bc Lumped_Flux For Cvfem_Turbulence_Dissipation_Rate ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Lumped_Flux For Cvfem_Turbulence_Dissipation_Rate ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f	"string"	–

User_Field

Syntax

Bc Lumped_Flux For Cvfem_Turbulence_Dissipation_Rate ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc Lumped_Flux For Cvfem_Turbulence_Dissipation_Rate ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc Lumped_Flux For Cvfem_Turbulence_Dissipation_Rate ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

Cvfem_Turbulent_Kinetic_Energy

Constant

Syntax

Bc Lumped_Flux For Cvfem_Turbulent_Kinetic_Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | Value = *value*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
value	real	–

Exponential

Syntax

Bc Lumped_Flux For Cvfem_Turbulent_Kinetic_Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Free_Open_Flow

Syntax

Bc Lumped_Flux For Cvfem_Turbulent_Kinetic_Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Free_Open_Flow [Toggle = *toggle*]

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–

Global

Syntax

Bc Lumped_Flux For Cvfem_Turbulent_Kinetic_Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
global_name	“string”	–
component	integer	0

Mass_Flux

Syntax

Bc Lumped_Flux For Cvfem_Turbulent_Kinetic_Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Mass_Flux ([Toggle = *toggle*] | Total_Mdot = *total_mdot* | [Inflow_Phi = *inflow_phi*])

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
total_mdot	real	–
inflow_phi	real	0

Nd_Table

Syntax

Bc Lumped_Flux For Cvfem_Turbulent_Kinetic_Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Open_Adv_Flow

Syntax

Bc Lumped_Flux For Cvfem_Turbulent_Kinetic_Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Open_Adv_Flow ([Toggle = *toggle*] | [Pressure = *pressure*] | [Total_Pressure = *total_pressure*] | [Far_Field_Entrainment_Value = *far_field_entrainment_value*])

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
pressure	real	0
total_pressure	real	0
far_field_entrainment_value	real	0

Polynomial

Syntax

Bc Lumped_Flux For Cvfem_Turbulent_Kinetic_Energy ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Lumped_Flux For Cvfem_Turbulent_Kinetic_Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f	"string"	–

User_Field

Syntax

Bc Lumped_Flux For Cvfem_Turbulent_Kinetic_Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc Lumped_Flux For Cvfem_Turbulent_Kinetic_Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc Lumped_Flux For Cvfem_Turbulent_Kinetic_Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

Cvfem_Velocity_Edge_Gradient_Projection

Constant

Syntax

Bc Lumped_Flux For Cvfem_Velocity_Edge_Gradient_Projection ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | Value = *value*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
value	real	–

Exponential

Syntax

Bc Lumped_Flux For Cvfem_Velocity_Edge_Gradient_Projection ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Lumped_Flux For Cvfem_Velocity_Edge_Gradient_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Bc Lumped_Flux For Cvfem_Velocity_Edge_Gradient_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc Lumped_Flux For Cvfem_Velocity_Edge_Gradient_Projection ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Lumped_Flux For Cvfem_Velocity_Edge_Gradient_Projection ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f	"string"	–

User_Field

Syntax

Bc Lumped_Flux For Cvfem_Velocity_Edge_Gradient_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc Lumped_Flux For Cvfem_Velocity_Edge_Gradient_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc Lumped_Flux For Cvfem_Velocity_Edge_Gradient_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

Cvfem_Velocity_Pressure_Projection

Constant

Syntax

Bc Lumped_Flux For Cvfem_Velocity_Pressure_Projection ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | [Valuex = *valuex*] | [Valuey = *valuey*] | [Valuez = *valuez*])

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
valuex	real	0
valuey	real	0
valuez	real	0

Simple_Dp_Interp

Syntax

Bc Lumped_Flux For Cvfem_Velocity_Pressure_Projection ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Simple_Dp_Interp [Toggle = *toggle*]

Summary

Simple DP interpolation model for CVFEM velocity pressure projection flux

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–

Vector_String_Function

Syntax

Bc Lumped_Flux For Cvfem_Velocity_Pressure_Projection ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Vector_String_Function ([Toggle = *toggle*] | F_X = *f_x* | F_Y = *f_y* | [F_Z = *f_z*])

Summary

Value from a string function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
f_x	“string”	–
f_y	“string”	–
f_z	“string”	–

Cvfem_Velocity_Pressure_Projection_X

Constant

Syntax

Bc Lumped_Flux For Cvfem_Velocity_Pressure_Projection_X ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | Value = *value*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
value	real	–

Exponential

Syntax

Bc Lumped_Flux For Cvfem_Velocity_Pressure_Projection_X ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Lumped_Flux For Cvfem_Velocity_Pressure_Projection_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Bc Lumped_Flux For Cvfem_Velocity_Pressure_Projection_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc Lumped_Flux For Cvfem_Velocity_Pressure_Projection_X ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Lumped_Flux For Cvfem_Velocity_Pressure_Projection_X ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f	"string"	–

User_Field

Syntax

Bc Lumped_Flux For Cvfem_Velocity_Pressure_Projection_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc Lumped_Flux For Cvfem_Velocity_Pressure_Projection_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc Lumped_Flux For Cvfem_Velocity_Pressure_Projection_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

Cvfem_Velocity_Pressure_Projection_Y

Constant

Syntax

Bc Lumped_Flux For Cvfem_Velocity_Pressure_Projection_Y ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | Value = *value*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
value	real	–

Exponential

Syntax

Bc Lumped_Flux For Cvfem_Velocity_Pressure_Projection_Y ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Lumped_Flux For Cvfem_Velocity_Pressure_Projection_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Bc Lumped_Flux For Cvfem_Velocity_Pressure_Projection_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc Lumped_Flux For Cvfem_Velocity_Pressure_Projection_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Lumped_Flux For Cvfem_Velocity_Pressure_Projection_Y ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f	"string"	–

User_Field

Syntax

Bc Lumped_Flux For Cvfem_Velocity_Pressure_Projection_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc Lumped_Flux For Cvfem_Velocity_Pressure_Projection_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc Lumped_Flux For Cvfem_Velocity_Pressure_Projection_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

Cvfem_Velocity_Pressure_Projection_Z

Constant

Syntax

Bc Lumped_Flux For Cvfem_Velocity_Pressure_Projection_Z ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | Value = *value*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
value	real	–

Exponential

Syntax

Bc Lumped_Flux For Cvfem_Velocity_Pressure_Projection_Z ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Lumped_Flux For Cvfem_Velocity_Pressure_Projection_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Bc Lumped_Flux For Cvfem_Velocity_Pressure_Projection_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc Lumped_Flux For Cvfem_Velocity_Pressure_Projection_Z ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Lumped_Flux For Cvfem_Velocity_Pressure_Projection_Z ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f	"string"	–

User_Field

Syntax

Bc Lumped_Flux For Cvfem_Velocity_Pressure_Projection_Z ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc Lumped_Flux For Cvfem_Velocity_Pressure_Projection_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc Lumped_Flux For Cvfem_Velocity_Pressure_Projection_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

Darcy_Momentum

Constant

Syntax

Bc Lumped_Flux For Darcy_Momentum ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | [Valuex = *valuex*] | [Valuey = *valuey*] | [Valuez = *valuez*])

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
valuex	real	0
valuey	real	0
valuez	real	0

Open_Flow

Syntax

Bc Lumped_Flux For Darcy_Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Open_Flow ([Toggle = *toggle*] | [Pressure = *pressure*] | [Total_Pressure = *total_pressure*])

Summary

Open flow momentum flux

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
pressure	real	0
total_pressure	real	0

Pressure

Syntax

Bc Lumped_Flux For Darcy_Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Pressure ([Toggle = *toggle*] | [User_Field_Name = *user_field_name*] | [Use_Bulk_Node = *use_bulk_node*] | [P = *p*] | [C_T = *c_t*] | [C_X = *c_x*] | [C_Y = *c_y*] | [C_Z = *c_z*] | [C_Cos_W = *c_cos_w*] | [C_Sin_W = *c_sin_w*] | [C1_Cos = *c1_cos*] | [C1_Sin = *c1_sin*])

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
user_field_name	"string"	–
use_bulk_node	"string"	–
p	real	0
c_t	real	0
c_x	real	0
c_y	real	0
c_z	real	0
c_cos_w	real	0
c_sin_w	real	0
c1_cos	real	0
c1_sin	real	0

Rt_Pressure

Syntax

Bc Lumped_Flux For Darcy_Momentum ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Rt_Pressure ([Toggle = *toggle*] | P = *p* | [C_T = *c_t*] | [C_X = *c_x*] | [C_Y = *c_y*] | [C_Z = *c_z*])

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
p	real	–
c_t	real	0
c_x	real	0
c_y	real	0
c_z	real	0

Simple_Interp

Syntax

Bc Lumped_Flux For Darcy_Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Simple_Interp [Toggle = *toggle*]

Summary

Darcy momentum simple interpolation flux BC

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–

Slip

Syntax

Bc Lumped_Flux For Darcy_Momentum ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Slip ([Toggle = *toggle*] | [Vs_X = *vs_x*] | [Vs_Y = *vs_y*] | [Vs_Z = *vs_z*] | K = *k*)

Summary

Darcy slip momentum flux

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
vs_x	real	0
vs_y	real	0
vs_z	real	0
k	real	–

Vector_String_Function

Syntax

Bc Lumped_Flux For Darcy_Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Vector_String_Function ([Toggle = *toggle*] | F_X = *f_x* | F_Y = *f_y* | [F_Z = *f_z*])

Summary

Value from a string function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
f_x	“string”	–
f_y	“string”	–
f_z	“string”	–

Darcy_Momentum_X

Constant

Syntax

Bc Lumped_Flux For Darcy_Momentum_X ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | Value = *value*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
value	real	–

Exponential

Syntax

Bc Lumped_Flux For Darcy_Momentum_X ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Lumped_Flux For Darcy_Momentum_X ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Bc Lumped_Flux For Darcy_Momentum_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc Lumped_Flux For Darcy_Momentum_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Lumped_Flux For Darcy_Momentum_X ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f	"string"	–

User_Field

Syntax

Bc Lumped_Flux For Darcy_Momentum_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc Lumped_Flux For Darcy_Momentum_X ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc Lumped_Flux For Darcy_Momentum_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

Darcy_Momentum_Y

Constant

Syntax

Bc Lumped_Flux For Darcy_Momentum_Y ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | Value = *value*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
value	real	–

Exponential

Syntax

Bc Lumped_Flux For Darcy_Momentum_Y ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Lumped_Flux For Darcy_Momentum_Y ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Bc Lumped_Flux For Darcy_Momentum_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc Lumped_Flux For Darcy_Momentum_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Lumped_Flux For Darcy_Momentum_Y ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f	"string"	–

User_Field

Syntax

Bc Lumped_Flux For Darcy_Momentum_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
scaling	real	1
global_var	“string”	–
time_function	“string”	–

User_Function

Syntax

Bc Lumped_Flux For Darcy_Momentum_Y ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc Lumped_Flux For Darcy_Momentum_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

Darcy_Momentum_Z

Constant

Syntax

Bc Lumped_Flux For Darcy_Momentum_Z ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | Value = *value*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
value	real	–

Exponential

Syntax

Bc Lumped_Flux For Darcy_Momentum_Z ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Lumped_Flux For Darcy_Momentum_Z ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Bc Lumped_Flux For Darcy_Momentum_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc Lumped_Flux For Darcy_Momentum_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Lumped_Flux For Darcy_Momentum_Z ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f	"string"	–

User_Field

Syntax

Bc Lumped_Flux For Darcy_Momentum_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc Lumped_Flux For Darcy_Momentum_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc Lumped_Flux For Darcy_Momentum_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

Div_Projection

Constant

Syntax

Bc Lumped_Flux For Div_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | [Valuex = *valuex*] | [Valuey = *valuey*] | [Valuez = *valuez*])

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
valuex	real	0
valuey	real	0
valuez	real	0

Vector_String_Function

Syntax

Bc Lumped_Flux For Div_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Vector_String_Function ([Toggle = *toggle*] | F_X = *f_x* | F_Y = *f_y* | [F_Z = *f_z*])

Summary

Value from a string function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f_x	"string"	–
f_y	"string"	–
f_z	"string"	–

Div_Projection_X

Constant

Syntax

Bc Lumped_Flux For Div_Projection_X ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | Value = *value*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
value	real	–

Exponential

Syntax

Bc Lumped_Flux For Div_Projection_X ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Lumped_Flux For Div_Projection_X ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Bc Lumped_Flux For Div_Projection_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc Lumped_Flux For Div_Projection_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Lumped_Flux For Div_Projection_X ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f	"string"	–

User_Field

Syntax

Bc Lumped_Flux For Div_Projection_X ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc Lumped_Flux For Div_Projection_X ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc Lumped_Flux For Div_Projection_X ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

Div_Projection_Y

Constant

Syntax

Bc Lumped_Flux For Div_Projection_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | Value = *value*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
value	real	–

Exponential

Syntax

Bc Lumped_Flux For Div_Projection_Y ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Lumped_Flux For Div_Projection_Y ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Bc Lumped_Flux For Div_Projection_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc Lumped_Flux For Div_Projection_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Lumped_Flux For Div_Projection_Y ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f	"string"	–

User_Field

Syntax

Bc Lumped_Flux For Div_Projection_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc Lumped_Flux For Div_Projection_Y ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc Lumped_Flux For Div_Projection_Y ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

Div_Projection_Z

Constant

Syntax

Bc Lumped_Flux For Div_Projection_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | Value = *value*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
value	real	–

Exponential

Syntax

Bc Lumped_Flux For Div_Projection_Z ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Lumped_Flux For Div_Projection_Z ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Bc Lumped_Flux For Div_Projection_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc Lumped_Flux For Div_Projection_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Lumped_Flux For Div_Projection_Z ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f	"string"	–

User_Field

Syntax

Bc Lumped_Flux For Div_Projection_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc Lumped_Flux For Div_Projection_Z ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc Lumped_Flux For Div_Projection_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
Name	“string”	–
plugin_parameters	“string”	–

Energy

Calore_User_Sub

Syntax

Bc Lumped_Flux For Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Calore_User_Sub ([Power_Output = *power_output*] | [Flux_Output = *flux_output*] | [Toggle = *toggle*] | Name = *name* | Type = *type* | [Multiplier = *multiplier*] | [Material_Data_Block = *material_data_block*] | [Data = *data*] | [Scaling_Field = *scaling_field*])

Summary

Values from a Calore user subroutine

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output

If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Flux_Output

If specified, the average flux of the given BC/Src is calculated and stored into the global variable name specified as the *flux_output* argument.

Toggle

If given, specifies which *toggle block* controls whether this BC is

active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
power_output	"string"	–
flux_output	"string"	–
toggle	"string"	–
name	"string"	–
type	"string"	–
multiplier	real	1
material_data_block	"string"	–
data	"string"	–
scaling_field	"string"	–

Cht_Flux

Syntax

Bc Lumped_Flux For Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Cht_Flux ([Power_Output = *power_output*] | [Flux_Output = *flux_output*] | [Toggle = *toggle*] | [Too_Field = *Too_Field*] | [H_Field = *H_Field*])

Summary

Applies H(T-Too) flux for conjugate heat transfer boundary. Optional parameters are "Too_Field" and "H_Field", which default to "nodal_cht_Too" and "nodal_cht_H" respectively.

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output

If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Flux_Output

If specified, the average flux of the given BC/Src is calculated and stored into the global variable name specified as the *flux_output* argument.

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
power_output	"string"	–
flux_output	"string"	–
toggle	"string"	–
Too_Field	"string"	nodal_cht_Too
H_Field	"string"	nodal_cht_H

Cht_Robin

Syntax

Bc Lumped_Flux For Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Cht_Robin ([Power_Output = *power_output*] | [Flux_Output = *flux_output*] | [Toggle = *toggle*])

Summary

Applies a Robin-style BC of the form: J dot n + h * (T-Tref).

Description

J, h, and Tref should be specified using the material properties BC_HEAT_FLUX, HEAT_TRANSFER_COEFFICIENT, and BC_REFERENCE_TEMPERATURE in a material that is applied to the surface.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output

If specified, the total (integrated) power of the given BC/Src is

calculated and stored into the global variable name specified as the *power_output* argument.

Flux_Output

If specified, the average flux of the given BC/Src is calculated and stored into the global variable name specified as the *flux_output* argument.

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
power_output	"string"	–
flux_output	"string"	–
toggle	"string"	–

Constant

Syntax

Bc Lumped_Flux For Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Power_Output = *power_output*] | [Flux_Output = *flux_output*] | [Toggle = *toggle*] | *{flux | value}* = *flux*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output

If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Flux_Output

If specified, the average flux of the given BC/Src is calculated and stored into the global variable name specified as the *flux_output* argument.

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
power_output	"string"	–
flux_output	"string"	–
toggle	"string"	–
flux	real	–

Distribution_Factor

Syntax

Bc Lumped_Flux For Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Distribution_Factor ([Power_Output = *power_output*] | [Flux_Output = *flux_output*] | [Toggle = *toggle*] | [Attribute_Name = *attribute_name*] | [Multiplier = *multiplier*])

Summary

Applies a flux equal to a distribution factor defined in the input mesh with name ATTRIBUTE_NAME. F = MULTIPLIER x ATTRIBUTE_NAME[node]

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output

If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Flux_Output

If specified, the average flux of the given BC/Src is calculated and stored into the global variable name specified as the *flux_output* argument.

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
power_output	"string"	–
flux_output	"string"	–
toggle	"string"	–
attribute_name	"string"	DISTRIBUTION_FACTOR
multiplier	real	1

Enclosure_Radiation

Syntax

Bc Lumped_Flux For Energy ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Enclosure_Radiation ([Power_Output = *power_output*] | [Flux_Output = *flux_output*] | [Toggle = *toggle*] | [Multiplier = *multiplier*] | Enclosure = *enclosure*)

Summary

This term imposes an energy flux due to enclosure radiation as defined in *Enclosure Radiation*. The irradiation is provided by the Chapparal radiosity solution

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output

If specified, the total (integrated) power of the given BC/Src is

calculated and stored into the global variable name specified as the *power_output* argument.

Flux_Output

If specified, the average flux of the given BC/Src is calculated and stored into the global variable name specified as the *flux_output* argument.

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
power_output	"string"	–
flux_output	"string"	–
toggle	"string"	–
multiplier	real	1
enclosure	"string"	–

Exponential

Syntax

Bc Lumped_Flux For Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Power_Output = *power_output*] | [Flux_Output = *flux_output*] | [Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

$$F = C + Me^{kX}$$

where X is the variable, C is the constant, M is the multiplier, and k is exponent. C and M are optional and default to 0 and 1 respectively.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output

If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Flux_Output

If specified, the average flux of the given BC/Src is calculated and stored into the global variable name specified as the *flux_output* argument.

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
power_output	"string"	–
flux_output	"string"	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Exponential_Vapor_Cooling

Syntax

Bc Lumped_Flux For Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential_Vapor_Cooling ([Power_Output = *power_output*] | [Flux_Output = *flux_output*] | [Toggle = *toggle*] | Tboil = *tboil* | L = *l* | Pa = *pa* | Mw = *mw* | Alpha = *alpha* | [Tl = *tl*] | [Th = *th*])

Summary

Exponential vapor cooling energy flux

Description

Flux is given by:

$$F = P_a L\alpha \exp\left(\frac{L}{R}\left(\frac{1}{T_{boil}} - \frac{1}{T}\right)\right)\sqrt{\frac{1}{2\pi RT}}$$

where R is the specific gas constant ($\bar{R}$/MW). For T less than T_L, zero flux is applied. For T between T_L and T_H, a 3rd order smoothing polynomial is used to provide continuity of values and derivatives between 0 and the expression given above. For T greater than T_H, the above expression is used. If not provided, T_L and T_H are set to 10% below and above T_{Boil}, respectively.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output

If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Flux_Output

If specified, the average flux of the given BC/Src is calculated and stored into the global variable name specified as the *flux_output* argument.

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
power_output	"string"	–
flux_output	"string"	–
toggle	"string"	–
tboil	real	–
l	real	–
pa	real	–
mw	real	–
alpha	real	–
tl	real	-1
th	real	-1

Face_Field_Scalar

Syntax

Bc Lumped_Flux For Energy ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Face_Field_Scalar ([Power_Output = *power_output*] | [Flux_Output = *flux_output*] | [Toggle = *toggle*] | Name = *name* | [Multiplier = *multiplier*] | [Advective_Bar = *advective_bar*] | [Command_Block_Name = *command_block_name*])

Summary

FACE or EDGE Field energy flux

Description

Flux is given by:

$$M \cdot f$$

M is given by MULTIPLIER and f is a face or edge field with name NAME.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output

If specified, the total (integrated) power of the given BC/Src is

calculated and stored into the global variable name specified as the *power_output* argument.

Flux_Output

If specified, the average flux of the given BC/Src is calculated and stored into the global variable name specified as the *flux_output* argument.

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
power_output	"string"	–
flux_output	"string"	–
toggle	"string"	–
name	"string"	–
multiplier	real	1
advective_bar	"string"	–
command_block_name	"string"	–

Fortran

Syntax

Bc Lumped_Flux For Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Fortran ([Power_Output = *power_output*] | [Flux_Output = *flux_output*] | [Toggle = *toggle*] | [Multiplier = *multiplier*] | Sub_Name = *sub_name* | [Real_Data = *real_data*] | [Int_Data = *int_data*] | [Resource_Name = *resource_name*] | [Data = *data*])

Summary

Fortran subroutine energy flux

Description

The following options are present in most boundary conditions/sources. The

meaning of each option is as follows:

Power_Output
If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Flux_Output
If specified, the average flux of the given BC/Src is calculated and stored into the global variable name specified as the *flux_output* argument.

Toggle
If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
power_output	“string”	–
flux_output	“string”	–
toggle	“string”	–
multiplier	real	1
sub_name	“string”	–
real_data	“string”	–
int_data	“string”	–
resource_name	“string”	–
data	“string”	–

Gaussian_Power_Weld

Syntax

Bc Lumped_Flux For Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Gaussian_Power_Weld ([Power_Output = *power_output*] | [Flux_Output = *flux_output*] | [Toggle = *toggle*] | R = *r* | [Clip_Radius = *clip_radius*] | [Efficiency = *efficiency*] | R_Eff = *R_eff* | Origin_And_Normal_Function =

origin_and_normal_function | [Old_Temperature = *old_temperature*] | Path_Function = *path_function* | [Deposition = *deposition*] | [Integrated = *integrated*] | [Clip = *clip*])

Summary

Energy flux due to an incident Gaussian laser beam

Description

The coordinates of the center of the beam are provided by the X, Y, and Z components of the provided origin_and_normal_function. The direction of the beam is provided by the NX, NY, and NZ components of the origin_and_normal_function. The applied flux is given by:

$$\frac{(-\boldsymbol{n})\cdot\boldsymbol{d}}{|\boldsymbol{d}|}\frac{\alpha e R_{eff} P}{\pi R^2(1-e^{-R_{eff}})}e^{-R_{eff}\frac{r_p^2}{R^2}}$$

P is the laser power defined for the material, R is the beam radius, R_{eff} is the effective beam radius, α is the laser absorptivity defined for the material, e is the EFFICIENCY, and r_p is the shortest distance between a point on the surface and the line defined by the beam origin and normal. The term $-\boldsymbol{n}\cdot\boldsymbol{d}$ accounts for the fact that the surface may not be orthogonal to the laser. Points with values of $r_p >$ CLIP_RADIUS will receive zero flux. If ACTIVATION is set to true, points whose temperature exceed ACTIVATION_TEMPERATURE will be flagged as activated for use with the ACTIVATION USER FUNCTION thermal conductivity model. If INTEGRATED is true, this flux will be integrated between the previous time step and the current one and the average flux applied.

See the *Directed Energy* section for the appropriate syntax to use this flux in the directed energy block.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output

If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Flux_Output

If specified, the average flux of the given BC/Src is calculated and stored into the global variable name specified as the *flux_output* argument.

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
power_output	"string"	–
flux_output	"string"	–
toggle	"string"	–
r	real	–
clip_radius	real	0
efficiency	real	1
R_eff	real	–
origin_and_normal_function	"string"	–
old_temperature	"string"	TRUE
path_function	"string"	–
deposition	"string"	FALSE
integrated	"string"	FALSE
clip	"string"	TRUE

Gaussian_Ray_Trace

Syntax

Bc Lumped_Flux For Energy ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Gaussian_Ray_Trace ([Power_Output = *power_output*] | [Flux_Output = *flux_output*] | [Toggle = *toggle*] | Path_Function = *path_function* | Dir = *dir* | Power = *power* | Type = *type* | [N_Rays = *n_rays*] | [N_R = *n_r*] | [N_Theta_Initial = *n_theta_initial*] | [N_Intersections = *n_intersections*] | [Weight_Ratio = *weight_ratio*] | [Id = *id*] | [Absorption_Model = *absorption_model*] | [Parallel_Type = *parallel_type*] | R = *r* | R_Eff = *r_eff*)

Summary

Ray tracing implementation of a Gaussian Beam energy flux

Description

r - Laser Gaussian standard deviation.

r_eff - Laser exponential factor. Gaussian is given by $EXP\left(\frac{-r_eff*D^2}{r^2}\right)$, where D

is the distance between a point and the laser center.

type - Either radial discretization or Monte Carlo. Determines how beam will be discretized into rays. Radial discretization is deterministic and breaks beams into approximately equal area segments radially. Each segment generates one ray. The power of each ray varies based on which segment it is generated from. Monte Carlo is stochastic and will generate a user-specified number of rays distributed spatially according to the laser's shape function, with each ray having equal power of power/n_rays.

n_r - Number of segments in radial direction for radial discretization.

n_theta_initial - Number of segments in theta direction in the smallest radial segment. The number of theta segments will grow as radial direction increases so segments have approximately the same area. Total number of rays will be n_r*n_r*n_theta_initial.

n_rays - Number of rays to generate for Monte Carlo discretization.

path_function - Function giving x, y, z coordinates of laser center as function of time.

dir - Function giving x,y,z components of direction that rays will travel initially from laser center defined by path_function.

power - Function giving total laser power.

n_intersections - Maximum number of times to intersect a ray with mesh elements due to reflections (default: 10).

weight_ratio - Fraction of initial ray power below which rays will be considered spent and no longer be traced (default: 0.001).

absorption_model - Either Specular, to use a surface emissivity value to determine fraction of energy absorbed and reflected with each ray intersection or Fresnel to use Fresnel model dependent on ray angle of incidence.

parallel_type - Either ray_parallel, to divide rays between processors with a globally shared mesh part or scene_parallel, to divide mesh part between processors with a globally shared set of rays.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output
: If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Flux_Output
: If specified, the average flux of the given BC/Src is calculated and

stored into the global variable name specified as the *flux_output* argument.

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
power_output	“string”	–
flux_output	“string”	–
toggle	“string”	–
path_function	“string”	–
dir	“string”	–
power	“string”	–
type	“string”	–
n_rays	integer	0
n_r	integer	0
n_theta_initial	integer	0
n_intersections	integer	10
weight_ratio	real	0.001
id	“string”	–
absorption_model	“string”	SPECULAR
parallel_type	“string”	RAY_PARALLEL
r	real	–
r_eff	real	–

Generalized_Nat_Conv

Syntax

Bc Lumped_Flux For Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Generalized_Nat_Conv ([Power_Output = *power_output*] | [Flux_Output = *flux_output*] | [Toggle = *toggle*] | [Multiplier = *multiplier*] | [H = *h*] | [T_Ref = *t_ref*])

Summary

Generalized natural convection energy flux

Description

Flux is given by:

$$F = Mh\left(T - T_{ref}\right)$$

M is given by MULTIPLIER, T_{ref} and h are taken from the BC REFERENCE TEMPERATURE and HEAT TRANSFER COEFFICIENT expressions.

The values for h and T_{ref} can also be supplied in-line as either constants or string functions.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output

If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Flux_Output

If specified, the average flux of the given BC/Src is calculated and stored into the global variable name specified as the *flux_output* argument.

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
power_output	“string”	–
flux_output	“string”	–
toggle	“string”	–
multiplier	real	1
h	“string”	–
t_ref	“string”	–

Generalized_Rad

Syntax

Bc Lumped_Flux For Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Generalized_Rad ([Power_Output = *power_output*] | [Flux_Output = *flux_output*] | [Toggle = *toggle*] | [Multiplier = *multiplier*] | [T_Ref = *t_ref*])

Summary

Radiative energy flux, with values for emissivity, form factor and irradiation taken from expressions

Description

Flux is:

$$M \epsilon f \left(\sigma T^4 - I \right)$$

M is MULTIPLIER. σ is the Stefan-Boltzmann constant. ϵ and F are expressions for emissivity and form factor. T is surface temperature. I is the expression for irradiation. If I does not already exist, it will be constructed as σT_{ref}^4, where T_{ref} is either provided by the BC Rad Reference Temperature expression or the inline T_{ref} parameter (constant or string function).

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output

If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Flux_Output

If specified, the average flux of the given BC/Src is calculated and stored into the global variable name specified as the *flux_output* argument.

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
power_output	"string"	–
flux_output	"string"	–
toggle	"string"	–
multiplier	real	1
t_ref	"string"	–

Global

Syntax

Bc Lumped_Flux For Energy ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Power_Output = *power_output*] | [Flux_Output = *flux_output*] | [Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output

If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Flux_Output

If specified, the average flux of the given BC/Src is calculated and stored into the global variable name specified as the *flux_output* argument.

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
power_output	"string"	–
flux_output	"string"	–
toggle	"string"	–
global_name	"string"	–
component	integer	0

Kuntz

Syntax

Bc Lumped_Flux For Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Kuntz ([Power_Output = *power_output*] | [Flux_Output = *flux_output*] | [Toggle = *toggle*] | [Multiplier = *multiplier*] | File = *file* | Units = *units*)

Summary

Interpolated value from tabulated data for the energy flux. FILE provides the filename for a table which specifies a flux value given X, Y, and Z coordinates and a temperature. UNITS specifies the units the table is in - ENGLISH ($\frac{BTU}{sft^2}$) or METRIC($\frac{W}{m^2}$)

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output

If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Flux_Output

If specified, the average flux of the given BC/Src is calculated and stored into the global variable name specified as the *flux_output* argument.

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
power_output	"string"	–
flux_output	"string"	–
toggle	"string"	–
multiplier	real	1
file	"string"	–
units	"string"	–

Lens

Syntax

Bc Lumped_Flux For Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Lens ([Power_Output = *power_output*] | [Flux_Output = *flux_output*] | [Toggle = *toggle*] | Src_X = *src_x* | Dir_X = *dir_x* | Vel_X = *vel_x* | Src_Y = *src_y* | Dir_Y = *dir_y* | Vel_Y = *vel_y* | [Src_Z = *src_z*] | [Dir_Z = *dir_z*] | [Vel_Z = *vel_z*] | Massflowrate = *massFlowRate* | T_Liq = *T_liq* | T_Solid = *T_solid* | [T_On = *t_on*] | [T_Off = *t_off*] | Masssourceradius = *massSourceRadius* | Particletemperature = *particleTemperature* | Particledensity = *particleDensity*)

Summary

Energy flux for modeling the LENS process

Description

This boundary condition imposes an energy flux due to an incident mass source representing a powder particle stream due to the LENS process. The source is at the coordinates

$$\boldsymbol{s} = \boldsymbol{x}_s + \boldsymbol{v}_s(t - T_ON)$$

where $\boldsymbol{x}_s$ is provided by the SRC vector coordinates and $\boldsymbol{v}_s$ is provided by the VEL vector coordinates. The source is directed from by the DIR vector (). The

point-wise energy flux is computed using the equation below:

$$\frac{(-\boldsymbol{n})\cdot\boldsymbol{d}}{|\boldsymbol{d}|}\left(\rho c_p T\frac{\dot{m}}{\rho_p \pi R^2}-\frac{\dot{m}c_p T_p}{\pi R^2}\right)$$

ρ, c_p, and T are the density, specific heat, and temperature of the surface. R, $\dot{m}$, ρ_p, and T_p are given by the massSourceRadius, massFlowRate, particleDensity, and particleTemperature parameters. The term $-\boldsymbol{n}\cdot\boldsymbol{d}$ accounts for the fact that the surface may not be orthogonal to the source. For points where the minimum distance between the point and the line defined by and is greater than R, the flux will be set to zero. Additionally, if T is less than (T_liq+T_solid)/2, the flux will be set to zero. For times less than T_ON or greater than T_OFF, the flux will also be set to zero.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output
: If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Flux_Output
: If specified, the average flux of the given BC/Src is calculated and stored into the global variable name specified as the *flux_output* argument.

Toggle
: If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
power_output	“string”	–
flux_output	“string”	–
toggle	“string”	–
src_x	real	–
dir_x	real	–
vel_x	real	–
src_y	real	–
dir_y	real	–
vel_y	real	–
src_z	real	0
dir_z	real	0
vel_z	real	0
massFlowRate	real	–
T_liq	real	–
T_solid	real	–
t_on	real	0
t_off	real	1.798e+308
massSourceRadius	real	–
particleTemperature	real	–
particleDensity	real	–

Nat_Conv

Syntax

Bc Lumped_Flux For Energy ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nat_Conv ([Power_Output = *power_output*] | [Flux_Output = *flux_output*] | [Toggle = *toggle*] | H = *h* | T_Ref = *t_ref*)

Summary

Convective energy flux defined with constants specified for heat transfer coefficient and reference temperature.

Description

Flux is given by:

$$F = h\left(T - T_{ref}\right)$$

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output

If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Flux_Output

If specified, the average flux of the given BC/Src is calculated and stored into the global variable name specified as the *flux_output* argument.

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
power_output	"string"	–
flux_output	"string"	–
toggle	"string"	–
h	real	–
t_ref	real	–

Nd_Table

Syntax

Bc Lumped_Flux For Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Power_Output = *power_output*] | [Flux_Output = *flux_output*] | [Toggle =

toggle] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table.

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output

If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Flux_Output

If specified, the average flux of the given BC/Src is calculated and stored into the global variable name specified as the *flux_output* argument.

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
power_output	“string”	–
flux_output	“string”	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Pid_Controlled

Syntax

Bc Lumped_Flux For Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Pid_Controlled ([Power_Output = *power_output*] | [Flux_Output = *flux_output*] | [Toggle = *toggle*] | Max_Output = *Max_Output* | [P = *p*] | [I = *i*] | [D = *d*] | [T_I = *t_i*] | [T_D = *t_d*] | [Band = *Band*] | [Filter_Tau = *Filter_Tau*] | [Filter_Number = *Filter_Number*] | Setpoint_Function_Name = *Setpoint_Function_Name* | Control_Variable = *Control_Variable* | [Enable_Clamping = *enable_clamping*])

Summary

This flux term imposes an energy flux using a PID controller

Description

The controller setpoint vs. time is defined in a user function, and the controller feedback point uses a global variable (e.g. a data probe or reduction postprocessor).

The derivative term uses a filtered error value, using a low pass filter whose time constant is specified by the user. Like a heater, this is only capable of providing a positive influx. Even if the controller output is negative, the applied energy source will be truncated at 0. Likewise, the applied energy source cannot exceed the specified max source.

The nominal scaled PID controller output (between 0 and 1) is

$$u(t) = K_p e(t) + K_i \int_0^t e(s)ds + K_d \frac{\partial e_f}{\partial t}$$

where it is important to note that the K_p coefficient does not scale the others unless you have set it using the Band argument. In some PID implementations, the K_i and K_d coefficients are internally multiplied by K_p so care must be taken to provide the correct coefficients. This output $u(t)$ which is from 0 to 1 is scaled by the specified Max_Output value to produce the applied energy source. This means you **should not** include the output magnitude scaling in your P, I, or D parameters. If this controller is meant to recreate a controller that commands power directly, one must normalize by the max power.

When a derivative gain is provided, the derivative action is calculated using a filtered error to avoid derivative kick. The `Filter_Tau` argument can be used to specify the first-order derivative filter time constant τ, with a default value of 1 otherwise. Alternatively, one can provide the `Filter_Number` as argument $\tau = \frac{1}{N}$ where

$$K_d \frac{N}{1 + N\frac{1}{s}} = K_d \frac{1}{\tau + \frac{1}{s}} = \frac{K_d s}{\tau s + 1}$$

The P, I, and D coefficients can be specified in a number of ways:

- Just give a 'Band' argument. This calculates $P = 1/Band$ and uses the defaults of $I = 0.01P$ and $D = 0.0$
- Give a 'Band' argument and non-defaults for I and/or D. This still multiplies whatever you give for I and D by P.
- Give a 'P' argument instead of 'Band'. NOTE: in this mode the I and D values are not scaled by P.

If you provide both P and Band, P is used, the Band argument is ignored, and I and D are not scaled.

Alternatively, the integral and/or derivative gains can be provided using the integral time T_i (`T_i` argument) and derivative time T_d (`T_d` argument) so that $I = P/T_i$ and $D = P * T_d$ (regardless if P was set via band arg or not).

If `ENABLE_CLAMPING` is set to true, a clamping anti-windup algorithm is applied to the integral action of the controller.

The default values for I and D are 0.01 and 0 if not provided. The setpoint, controller output, error, filtered error, and filtered error time derivative are all output to automatically created global variables for diagnostics and postprocessing.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output
: If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Flux_Output
: If specified, the average flux of the given BC/Src is calculated and stored into the global variable name specified as the *flux_output* argument.

Toggle
: If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
power_output	"string"	–
flux_output	"string"	–
toggle	"string"	–
Max_Output	real	–
p	real	0
i	real	0
d	real	0
t_i	real	0
t_d	real	0
Band	real	0
Filter_Tau	real	1
Filter_Number	real	1
Setpoint_Function_Name	"string"	–
Control_Variable	"string"	–
enable_clamping	integer	–

Pid_Controlled_Bidirectional

Syntax

Bc Lumped_Flux For Energy ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Pid_Controlled_Bidirectional ([Power_Output = *power_output*] | [Flux_Output = *flux_output*] | [Toggle = *toggle*] | Max_Cooling_Power = *Max_Cooling_Power* | Max_Heating_Power = *Max_Heating_Power* | [P = *p*] | [I = *i*] | [D = *d*] | [T_I = *t_i*] | [T_D = *t_d*] | [Band = *Band*] | [Filter_Tau = *Filter_Tau*] | [Filter_Number = *Filter_Number*] | Setpoint_Function_Name = *Setpoint_Function_Name* | Control_Variable = *Control_Variable* | [Enable_Clamping = *enable_clamping*])

Summary

This is similar to PID_Controlled, however, the flux is applied to cool and heat the volume instead of just heating it. Max_Cooling_Power and Max_Heating_Power should be positive values. See description for PID_CONTROLLED for more details

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output

If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Flux_Output

If specified, the average flux of the given BC/Src is calculated and stored into the global variable name specified as the *flux_output* argument.

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
power_output	"string"	–
flux_output	"string"	–
toggle	"string"	–
Max_Cooling_Power	real	–
Max_Heating_Power	real	–
p	real	0
i	real	0
d	real	0
t_i	real	0
t_d	real	0
Band	real	0
Filter_Tau	real	1
Filter_Number	real	1
Setpoint_Function_Name	"string"	–
Control_Variable	"string"	–
enable_clamping	integer	–

Pid_Controlled_Cooler

Syntax

Bc Lumped_Flux For Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Pid_Controlled_Cooler ([Power_Output = *power_output*] | [Flux_Output = *flux_output*] | [Toggle = *toggle*] | [P = *p*] | [I = *i*] | [D = *d*] | [T_I = *t_i*] | [T_D = *t_d*] | [Band = *Band*] | [Filter_Tau = *Filter_Tau*] | [Filter_Number = *Filter_Number*] | Setpoint_Function_Name = *Setpoint_Function_Name* | Control_Variable = *Control_Variable* | [Enable_Clamping = *enable_clamping*] | Max_Cooling_Power = *Max_Cooling_Power*)

Summary

This is similar to PID_Controlled, however, the flux is applied to cool the volume instead of heat it. Max_Cooling_Power should be a positive value. See description for PID_CONTROLLED for more details

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output

If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Flux_Output

If specified, the average flux of the given BC/Src is calculated and stored into the global variable name specified as the *flux_output* argument.

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
power_output	"string"	–
flux_output	"string"	–
toggle	"string"	–
p	real	0
i	real	0
d	real	0
t_i	real	0
t_d	real	0
Band	real	0
Filter_Tau	real	1
Filter_Number	real	1
Setpoint_Function_Name	"string"	–
Control_Variable	"string"	–
enable_clamping	integer	–
Max_Cooling_Power	real	–

Polynomial

Syntax

Bc Lumped_Flux For Energy ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Power_Output = *power_output*] | [Flux_Output = *flux_output*] | [Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The flux, F, is calculated as $F = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The

meaning of each option is as follows:

Power_Output
: If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Flux_Output
: If specified, the average flux of the given BC/Src is calculated and stored into the global variable name specified as the *flux_output* argument.

Toggle
: If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
power_output	“string”	–
flux_output	“string”	–
toggle	“string”	–
variable	“string”	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Rad

Syntax

Bc Lumped_Flux For Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Rad ([Power_Output = *power_output*] | [Flux_Output = *flux_output*] | [Toggle = *toggle*] | Crad = *crad* | T_Ref = *t_ref*)

Summary

Radiative energy flux defined with constant models for emissivity and reference temperature.

Description

The applied flux is given by:

$$c_{rad}\left(T^4 - T_{ref}^4\right)$$

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output

If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Flux_Output

If specified, the average flux of the given BC/Src is calculated and stored into the global variable name specified as the *flux_output* argument.

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
power_output	"string"	–
flux_output	"string"	–
toggle	"string"	–
crad	real	–
t_ref	real	–

Ray_Trace

Syntax

Bc Lumped_Flux For Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Ray_Trace ([Power_Output = *power_output*] | [Flux_Output = *flux_output*] | [Toggle = *toggle*] | Ray_Source = *ray_source* | [Emission = *emission*])

Summary

Apply a ray tracing energy flux

Description

Adds an energy flux based on the intersection of rays defined in a ray tracing source block for the equation system with name given by the RAY_SOURCE parameter with the surface. The optional EMISSION parameter also imposes a σT^4 blackbody radiative emission term

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output

If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Flux_Output

If specified, the average flux of the given BC/Src is calculated and stored into the global variable name specified as the *flux_output* argument.

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
power_output	“string”	–
flux_output	“string”	–
toggle	“string”	–
ray_source	“string”	–
emission	“string”	FALSE

Rte_Field

Syntax

Bc Lumped_Flux For Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Rte_Field ([Power_Output = *power_output*] | [Flux_Output = *flux_output*] | [Toggle = *toggle*])

Summary

RTE energy flux from a field

Description

Flux is given by -f, where f is the field named ‘rte_radiative_flux’

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output

If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Flux_Output

If specified, the average flux of the given BC/Src is calculated and

stored into the global variable name specified as the *flux_output* argument.

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
power_output	"string"	–
flux_output	"string"	–
toggle	"string"	–

Rte_Sp

Syntax

Bc Lumped_Flux For Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Rte_Sp ([Power_Output = *power_output*] | [Flux_Output = *flux_output*] | [Toggle = *toggle*] | Order = *order*)

Summary

RTE flux for energy equation with Simplified Spherical Harmonic RTE model with order ORDER

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output

If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Flux_Output

If specified, the average flux of the given BC/Src is calculated and stored into the global variable name specified as the *flux_output*

argument.

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
power_output	"string"	–
flux_output	"string"	–
toggle	"string"	–
order	integer	–

Scalar_String_Function

Syntax

Bc Lumped_Flux For Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Power_Output = *power_output*] | [Flux_Output = *flux_output*] | [Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output

If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Flux_Output

If specified, the average flux of the given BC/Src is calculated and stored into the global variable name specified as the *flux_output* argument.

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
power_output	"string"	–
flux_output	"string"	–
toggle	"string"	–
f	"string"	–

Scaled_Nat_Conv

Syntax

Bc Lumped_Flux For Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scaled_Nat_Conv ([Power_Output = *power_output*] | [Flux_Output = *flux_output*] | [Toggle = *toggle*] | [Multiplier = *multiplier*] | Scaling_Field = *scaling_field* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Scaled natural convection energy flux

Description

Flux is given by:

$$(M)(SC)(GV)(f)fcn(t)h\left(T-T_{ref}\right)$$

M is given by MULTIPLIER, SC by SCALING, GV by the value of the global variable with name GLOBAL_VAR, fcn is a function with name TIME_FUNCTION, and f is a field with name SCALING_FIELD. h and T_{ref} are taken from the HEAT TRANSFER COEFFICIENT and BC REFERENCE TEMPERATURE expressions.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output

If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Flux_Output

If specified, the average flux of the given BC/Src is calculated and stored into the global variable name specified as the *flux_output* argument.

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
power_output	“string”	–
flux_output	“string”	–
toggle	“string”	–
multiplier	real	1
scaling_field	“string”	–
scaling	real	1
global_var	“string”	–
time_function	“string”	–

Sharp_Power_Weld

Syntax

Bc Lumped_Flux For Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Sharp_Power_Weld ([Power_Output = *power_output*] | [Flux_Output = *flux_output*] | [Toggle = *toggle*] | R = *r* | [Clip_Radius = *clip_radius*] | [Efficiency = *efficiency*] | Origin_And_Normal_Function = *origin_and_normal_function* | [Old_Temperature = *old_temperature*] | Path_Function = *path_function* | [Deposition = *deposition*] | [Integrated = *integrated*] | [Clip = *clip*])

Summary

Energy flux due to an incident 'top hat' beam

Description

The coordinates of the center of the beam are provided by the X, Y, and Z components of the provided origin_and_normal_function. The direction of the beam is provided by the NX, NY, and NZ components of the origin_and_normal_function. The applied flux is given by:

$$\frac{(-\boldsymbol{n})\cdot\boldsymbol{d}}{|\boldsymbol{d}|}\frac{\alpha e P}{\pi R^2}$$

P is the laser power defined for the material, R is the beam radius, α is the laser absorptivity defined for the material, e is the EFFICIENCY, and the term $-\boldsymbol{n}\cdot\boldsymbol{d}$ accounts for the fact that the surface may not be orthogonal to the laser. Points with values of r_p > CLIP_RADIUS will receive zero flux. r_p is the shortest distance between a point on the surface and the line defined by the beam origin and normal. If ACTIVATION is set to true, points whose temperature exceed ACTIVATION_TEMPERATURE will be flagged as activated for use with the ACTIVATION USER FUNCTION thermal conductivity model. If INTEGRATED is true, this flux will be integrated between the previous time step and the current one and the average flux applied.

See the *Directed Energy* section for the appropriate syntax to use this flux in the directed energy block.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output

If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Flux_Output

If specified, the average flux of the given BC/Src is calculated and stored into the global variable name specified as the *flux_output* argument.

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
power_output	"string"	–
flux_output	"string"	–
toggle	"string"	–
r	real	–
clip_radius	real	0
efficiency	real	1
origin_and_normal_function	"string"	–
old_temperature	"string"	TRUE
path_function	"string"	–
deposition	"string"	FALSE
integrated	"string"	FALSE
clip	"string"	TRUE

Sharp_Ray_Trace

Syntax

Bc Lumped_Flux For Energy ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Sharp_Ray_Trace ([Power_Output = *power_output*] | [Flux_Output = *flux_output*] | [Toggle = *toggle*] | Path_Function = *path_function* | Dir = *dir* | Power = *power* | Type = *type* | [N_Rays = *n_rays*] | [N_R = *n_r*] | [N_Theta_Initial = *n_theta_initial*] | [N_Intersections = *n_intersections*] | [Weight_Ratio = *weight_ratio*] | [Id = *id*] | [Absorption_Model = *absorption_model*] | [Parallel_Type = *parallel_type*] | R = *r*)

Summary

Ray tracing implementation of a sharp circular beam energy flux

Description

r - Laser spot radius.

type - Either radial discretization or Monte Carlo. Determines how beam will be discretized into rays. Radial discretization is deterministic and breaks beams into approximately equal area segments radially. Each segment generates one

ray. The power of each ray varies based on which segment it is generated from. Monte Carlo is stochastic and will generate a user-specified number of rays distributed spatially according to the laser's shape function, with each ray having equal power of power/n_rays.

n_r - Number of segments in radial direction for radial discretization.

n_theta_initial - Number of segments in theta direction in the smallest radial segment. The number of theta segments will grow as radial direction increases so segments have approximately the same area. Total number of rays will be n_r*n_r*n_theta_initial.

n_rays - Number of rays to generate for Monte Carlo discretization.

path_function - Function giving x, y, z coordinates of laser center as function of time.

dir - Function giving x,y,z components of direction that rays will travel initially from laser center defined by path_function.

power - Function giving total laser power.

n_intersections - Maximum number of times to intersect a ray with mesh elements due to reflections (default: 10).

weight_ratio - Fraction of initial ray power below which rays will be considered spent and no longer be traced (default: 0.001).

absorption_model - Either Specular, to use a surface emissivity value to determine fraction of energy absorbed and reflected with each ray intersection or Fresnel to use Fresnel model dependent on ray angle of incidence.

parallel_type - Either ray_parallel, to divide rays between processors with a globally shared mesh part or scene_parallel, to divide mesh part between processors with a globally shared set of rays.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output
: If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Flux_Output
: If specified, the average flux of the given BC/Src is calculated and stored into the global variable name specified as the *flux_output* argument.

Toggle
: If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
power_output	“string”	–
flux_output	“string”	–
toggle	“string”	–
path_function	“string”	–
dir	“string”	–
power	“string”	–
type	“string”	–
n_rays	integer	0
n_r	integer	0
n_theta_initial	integer	0
n_intersections	integer	10
weight_ratio	real	0.001
id	“string”	–
absorption_model	“string”	SPECULAR
parallel_type	“string”	RAY_PARALLEL
r	real	–

Thermal_Latent_Heat

Syntax

Bc Lumped_Flux For Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Thermal_Latent_Heat ([Power_Output = *power_output*] | [Flux_Output = *flux_output*] | [Toggle = *toggle*])

Summary

Thermal latent heat energy flux

Description

Flux is given by:

$$H_v v_e$$

H_v is the expression for latent heat of vaporization and v_e is the local interface speed

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output
If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Flux_Output
If specified, the average flux of the given BC/Src is calculated and stored into the global variable name specified as the *flux_output* argument.

Toggle
If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
power_output	"string"	–
flux_output	"string"	–
toggle	"string"	–

User_Field

Syntax

Bc Lumped_Flux For Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Power_Output = *power_output*] | [Flux_Output = *flux_output*] | [Toggle = *toggle*] | [Multiplier = *multiplier*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

User field energy flux

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output

If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Flux_Output

If specified, the average flux of the given BC/Src is calculated and stored into the global variable name specified as the *flux_output* argument.

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
power_output	“string”	–
flux_output	“string”	–
toggle	“string”	–
multiplier	real	1
name	“string”	–
scaling	real	1
global_var	“string”	–
time_function	“string”	–

User_Function

Syntax

Bc Lumped_Flux For Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Power_Output = *power_output*] | [Flux_Output = *flux_output*] | [Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier =

multiplier])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output

If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Flux_Output

If specified, the average flux of the given BC/Src is calculated and stored into the global variable name specified as the *flux_output* argument.

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
power_output	“string”	–
flux_output	“string”	–
toggle	“string”	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc Lumped_Flux For Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Power_Output = *power_output*] | [Flux_Output = *flux_output*] | [Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output

If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Flux_Output

If specified, the average flux of the given BC/Src is calculated and stored into the global variable name specified as the *flux_output* argument.

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
power_output	“string”	–
flux_output	“string”	–
toggle	“string”	–
Name	“string”	–
plugin_parameters	“string”	–

User_Vector_Field

Syntax

Bc Lumped_Flux For Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Vector_Field ([Power_Output = *power_output*] | [Flux_Output = *flux_output*] | [Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*] | [Multiplier = *multiplier*])

Summary

User vector field energy flux

Description

Flux is given by:

$$(M)(SC)(GV)fcn(t)\boldsymbol{f}\cdot\boldsymbol{n}$$

M is given by MULTIPLIER, SC by SCALING, GV by the value of the global variable with name GLOBAL_VAR, fcn is a function with name TIME_FUNCTION, f is a field with name NAME, and n is the surface normal

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output

If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Flux_Output

If specified, the average flux of the given BC/Src is calculated and stored into the global variable name specified as the *flux_output* argument.

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
power_output	“string”	–
flux_output	“string”	–
toggle	“string”	–
name	“string”	–
scaling	real	1
global_var	“string”	–
time_function	“string”	–
multiplier	real	1

User_Vector_Field_Influx

Syntax

Bc Lumped_Flux For Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Vector_Field_Influx ([Power_Output = *power_output*] | [Flux_Output = *flux_output*] | [Toggle = *toggle*] | [Multiplier = *multiplier*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

User vector field influx energy flux

Description

Flux is given by:

$$(M)(SC)(GV)fcn(t)\boldsymbol{f}\cdot\boldsymbol{n}$$

M is given by MULTIPLIER, SC by SCALING, GV by the value of the global variable with name GLOBAL_VAR, fcn is a function with name TIME_FUNCTION, f is a field with name NAME, and n is the surface normal. Flux will be capped at zero (inflow only)

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output

If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Flux_Output

If specified, the average flux of the given BC/Src is calculated and stored into the global variable name specified as the *flux_output* argument.

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
power_output	“string”	–
flux_output	“string”	–
toggle	“string”	–
multiplier	real	1
name	“string”	–
scaling	real	1
global_var	“string”	–
time_function	“string”	–

Vapor_Cooling

Syntax

Bc Lumped_Flux For Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Vapor_Cooling ([Power_Output = *power_output*] | [Flux_Output = *flux_output*] | [Toggle = *toggle*] | Tboil = *tboil*)

Summary

Empirical relation for cooling due to vaporization of stainless steel.

Description

Flux is given by:

$$814373(T - T_{boil}) - 2248.31(T - T_{boil})^2 + 27.1683(T - T_{boil})^3$$

For $T - T_{boil}$ less than 170 and:

$$-310360000 + 3.272400(T - T_{boil}) - 1808.4(T - T_{boil})^2 + 2.7284(T - T_{boil})^3$$

For $T - T_{boil}$ greater than 170. For $T - T_{boil}$ less than zero, no flux is applied

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output

If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Flux_Output

If specified, the average flux of the given BC/Src is calculated and stored into the global variable name specified as the *flux_output* argument.

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
power_output	“string”	–
flux_output	“string”	–
toggle	“string”	–
tboil	real	–

Enthalpy

Constant

Syntax

Bc Lumped_Flux For Enthalpy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | Value = *value*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
value	real	–

Convective_Outflow

Syntax

Bc Lumped_Flux For Enthalpy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Convective_Outflow [Toggle = *toggle*]

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–

Exponential

Syntax

Bc Lumped_Flux For Enthalpy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Generalized_Nat_Conv

Syntax

Bc Lumped_Flux For Enthalpy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Generalized_Nat_Conv ([Toggle = *toggle*] | [Multiplier = *multiplier*])

Summary

Generalized natural convection enthalpy flux

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
multiplier	real	1

Generalized_Rad

Syntax

Bc Lumped_Flux For Enthalpy ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Generalized_Rad ([Toggle = *toggle*] | [Multiplier = *multiplier*] | [T_Ref = *t_ref*])

Summary

Radiative energy flux, with values for emissivity, form factor and irradiation taken from expressions

Description

Flux is:

$$M\epsilon f\left(\sigma T^4 - I\right)$$

M is MULTIPLIER. σ is the Stefan-Boltzmann constant. ϵ and F are expressions for emissivity and form factor. T is surface temperature. I is the expression for irradiation. If I does not already exist, it will be constructed as σT_{ref}^4, where T_{ref} is either provided by the BC Rad Reference Temperature expression or the inline T_{ref} parameter (constant or string function).

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
multiplier	real	1
t_ref	“string”	–

Global

Syntax

Bc Lumped_Flux For Enthalpy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
global_name	"string"	–
component	integer	0

Nat_Conv

Syntax

Bc Lumped_Flux For Enthalpy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nat_Conv ([Toggle = *toggle*] | H = *h* | T_Ref = *t_ref*)

Summary

Natural convection BC for enthalpy flux

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
h	real	–
t_ref	real	–

Nd_Table

Syntax

Bc Lumped_Flux For Enthalpy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc Lumped_Flux For Enthalpy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
variable	“string”	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Rad

Syntax

Bc Lumped_Flux For Enthalpy ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Rad ([Toggle = *toggle*] | Crad = *crad* | T_Ref = *t_ref*)

Summary

Radiative energy flux defined with constant models for emissivity and reference temperature.

Description

The applied flux is given by:

$$c_{rad}\left(T^4 - T_{ref}^4\right)$$

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
crad	real	–
t_ref	real	–

Scalar_String_Function

Syntax

Bc Lumped_Flux For Enthalpy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f	"string"	–

User_Field

Syntax

Bc Lumped_Flux For Enthalpy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc Lumped_Flux For Enthalpy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc Lumped_Flux For Enthalpy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

Level_Set

Constant

Syntax

Bc Lumped_Flux For Level_Set ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | *{flux | value}* = *flux*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
flux	real	–

Exponential

Syntax

Bc Lumped_Flux For Level_Set ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Lumped_Flux For Level_Set ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
global_name	"string"	–
component	integer	0

Inflow_Outflow

Syntax

Bc Lumped_Flux For Level_Set ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Inflow_Outflow [Toggle = *toggle*]

Summary

This model has been deprecated and removed from the code.

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–

Nd_Table

Syntax

Bc Lumped_Flux For Level_Set ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc Lumped_Flux For Level_Set ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
variable	“string”	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Lumped_Flux For Level_Set ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
f	“string”	–

User_Field

Syntax

Bc Lumped_Flux For Level_Set ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
scaling	real	1
global_var	“string”	–
time_function	“string”	–

User_Function

Syntax

Bc Lumped_Flux For Level_Set ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc Lumped_Flux For Level_Set ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

Lumped_Div_Projection

Constant

Syntax

Bc Lumped_Flux For Lumped_Div_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | [Valuex = *valuex*] | [Valuey = *valuey*] | [Valuez = *valuez*])

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
valuex	real	0
valuey	real	0
valuez	real	0

Vector_String_Function

Syntax

Bc Lumped_Flux For Lumped_Div_Projection ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Vector_String_Function ([Toggle = *toggle*] | F_X = *f_x* | F_Y = *f_y* | [F_Z = *f_z*])

Summary

Value from a string function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f_x	"string"	–
f_y	"string"	–
f_z	"string"	–

Lumped_Div_Projection_X

Constant

Syntax

Bc Lumped_Flux For Lumped_Div_Projection_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | Value = *value*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
value	real	–

Exponential

Syntax

Bc Lumped_Flux For Lumped_Div_Projection_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Lumped_Flux For Lumped_Div_Projection_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
global_name	“string”	–
component	integer	0

Nd_Table

Syntax

Bc Lumped_Flux For Lumped_Div_Projection_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc Lumped_Flux For Lumped_Div_Projection_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Lumped_Flux For Lumped_Div_Projection_X ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f	"string"	–

User_Field

Syntax

Bc Lumped_Flux For Lumped_Div_Projection_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc Lumped_Flux For Lumped_Div_Projection_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc Lumped_Flux For Lumped_Div_Projection_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

Lumped_Div_Projection_Y

Constant

Syntax

Bc Lumped_Flux For Lumped_Div_Projection_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | Value = *value*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
value	real	–

Exponential

Syntax

Bc Lumped_Flux For Lumped_Div_Projection_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Lumped_Flux For Lumped_Div_Projection_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Bc Lumped_Flux For Lumped_Div_Projection_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc Lumped_Flux For Lumped_Div_Projection_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Lumped_Flux For Lumped_Div_Projection_Y ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f	"string"	–

User_Field

Syntax

Bc Lumped_Flux For Lumped_Div_Projection_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc Lumped_Flux For Lumped_Div_Projection_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc Lumped_Flux For Lumped_Div_Projection_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

Lumped_Div_Projection_Z

Constant

Syntax

Bc Lumped_Flux For Lumped_Div_Projection_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | Value = *value*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
value	real	–

Exponential

Syntax

Bc Lumped_Flux For Lumped_Div_Projection_Z ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Lumped_Flux For Lumped_Div_Projection_Z ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Bc Lumped_Flux For Lumped_Div_Projection_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc Lumped_Flux For Lumped_Div_Projection_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Lumped_Flux For Lumped_Div_Projection_Z ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f	"string"	–

User_Field

Syntax

Bc Lumped_Flux For Lumped_Div_Projection_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc Lumped_Flux For Lumped_Div_Projection_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc Lumped_Flux For Lumped_Div_Projection_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

Mass_Balance

Co2_Convective_Outflow

Syntax

Bc Lumped_Flux For Mass_Balance ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Co2_Convective_Outflow ([Toggle = *toggle*] | [Ref_Frac = *ref_frac*])

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
ref_frac	real	0

Constant

Syntax

Bc Lumped_Flux For Mass_Balance ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | Value = *value*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
value	real	–

Constant_Velocity

Syntax

Bc Lumped_Flux For Mass_Balance ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant_Velocity ([Toggle = *toggle*] | Mult = *Mult*)

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is

active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Mult	real	–

Convective_Outflow

Syntax

Bc Lumped_Flux For Mass_Balance ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Convective_Outflow ([Toggle = *toggle*] | [Ref_Frac = *ref_frac*])

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
ref_frac	real	0

Exponential

Syntax

Bc Lumped_Flux For Mass_Balance ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
variable	“string”	–
constant	real	0
multiplier	real	1
exponent	real	–

Fluid_Robin_Coupled_One_Region

Syntax

Bc Lumped_Flux For Mass_Balance ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Fluid_Robin_Coupled_One_Region ([Toggle = *toggle*] | [Coeff_Scaling = *coeff_scaling*])

Summary

Boundary condition for coupling of continuity to a free fluid in the same region.

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
coeff_scaling	real	0.001

Global

Syntax

Bc Lumped_Flux For Mass_Balance ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
global_name	"string"	–
component	integer	0

Mass_Fluid_Robin_Coupled

Syntax

Bc Lumped_Flux For Mass_Balance ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Mass_Fluid_Robin_Coupled ([Toggle = *toggle*] | [Coeff_Scaling = *coeff_scaling*] | P_Field = *P_Field* | Massflux_Field = *MassFlux_Field*)

Summary

Boundary condition for loose coupling of continuity to a free fluid region

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
coeff_scaling	real	0.001
P_Field	"string"	–
MassFlux_Field	"string"	–

Mass_Fraction_Fluid_Robin_Coupled

Syntax

Bc Lumped_Flux For Mass_Balance ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Mass_Fraction_Fluid_Robin_Coupled ([Toggle = *toggle*] | Mass_Fraction_Field = *mass_fraction_field* | Mass_Fraction_Diffusive_Flux_Field = *mass_fraction_diffusive_flux_field* | Coeff_Field = *coeff_field*)

Summary

Boundary condition for loose coupling of scalars to a free fluid region

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
mass_fraction_field	"string"	–
mass_fraction_diffusive_flux_field	"string"	–
coeff_field	"string"	–

Mass_Fraction_Fluid_Robin_Coupled_One_Region

Syntax

Bc Lumped_Flux For Mass_Balance ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Mass_Fraction_Fluid_Robin_Coupled_One_Region [Toggle = *toggle*]

Summary

Boundary condition for coupling of scalars to a free fluid in the same region.

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–

Mass_Open

Syntax

Bc Lumped_Flux For Mass_Balance ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Mass_Open ([Toggle = *toggle*] | [Pressure = *pressure*])

Summary

A weakly enforced pressure condition for open flow from a porous flow domain

Description

This model weakly enforces a pressure condition on the boundary of a porous domain, either equal to the specified constant `PRESSURE`, or set using the `BOUNDARY_PRESSURE` model from the surface material.

Note: This should only be used with porous flow equations.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
pressure	real	0

Mesh_Motion

Syntax

Bc Lumped_Flux For Mass_Balance ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Mesh_Motion [Toggle = *toggle*]

Summary

Moving boundary condition for Arbitrary Lagrangian-Eulerian (ALE) mesh motion in relative flux form. Accounts for the boundary advective flux due to general mesh motion.

Description

Applies the following boundary flux

$$-\int q\hat{\mathbf{u}}\cdot\mathbf{n}dS,$$

where $\hat{u}$ is the mesh velocity and where the integrated scalar q depends on the equation. For the gas phase `porous_enthalpy` equation with a `transported enthalpy = porous`, $q = \phi\rho_g h_g$. Similarly, for the gas phase `mass_balance` equation, $q = \phi\rho_g$ is the bulk mass density.

This flux boundary condition is automatically applied to the mass balance and porous enthalpy equations when the `use relative mass flux` porous flow option is turned on.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–

Moving_Boundary

Syntax

Bc Lumped_Flux For Mass_Balance ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Moving_Boundary [Toggle = *toggle*]

Summary

No penetration moving boundary condition, valid for Lagrangian mesh motion in non-relative flux form or relative flux form.

Description

Applies the following boundary flux

$$\int q\hat{\mathbf{u}} \cdot \mathbf{n} dS,$$

where $\hat{u}$ is the mesh velocity and where the integrated scalar q depends on the equation. If we consider the gas phase `porous_enthalpy` equation with a `transported enthalpy = porous`, then $q = \phi\rho_g h_g$. Similarly, for the gas phase `mass_balance` equation, $q = \phi\rho_g$ is the bulk mass density.

If we consider the gas phase `mass_balance` equation, with a gas phase porous velocity relative to the moving solid phase, then physically this boundary condition is derived from applying a no flux condition at moving boundary i.e.,

$$\mathbf{u}_g \cdot \mathbf{n} = \phi\hat{\mathbf{u}} \cdot \mathbf{n}.$$

In the Lagrangian setting, this flux is required to impose a no flux condition at the moving boundary with both the non-relative form (default) and relative form of the advection term. When using the relative form, typically a do-nothing condition would be sufficient to impose no flux; however, to simplify general ALE mesh motion for porous flow equations (i.e., mass balance and porous enthalpy) we automatically provide the mesh motion flux on boundaries for blocks that have mesh motion. Consequently, to impose a zero flux, this term is needed for cancellation.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–

Nd_Table

Syntax

Bc Lumped_Flux For Mass_Balance ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Open

Syntax

Bc Lumped_Flux For Mass_Balance ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Open ([Toggle = *toggle*] | [Pressure = *pressure*] | [Entrained_Value = *entrained_value*])

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
pressure	real	0
entrained_value	real	0

Open_Flow

Syntax

Bc Lumped_Flux For Mass_Balance ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Open_Flow [Toggle = *toggle*]

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–

Polynomial

Syntax

Bc Lumped_Flux For Mass_Balance ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
variable	“string”	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Lumped_Flux For Mass_Balance ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
f	“string”	–

User_Field

Syntax

Bc Lumped_Flux For Mass_Balance ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc Lumped_Flux For Mass_Balance ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc Lumped_Flux For Mass_Balance ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

Well_Outflow

Syntax

Bc Lumped_Flux For Mass_Balance ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Well_Outflow ([Toggle = *toggle*] | Pi = *pi* | Pref = *pref*)

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
pi	"string"	–
pref	"string"	–

Mesh

Constant

Syntax

Bc Lumped_Flux For Mesh ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | [Valuex = *valuex*] | [Valuey = *valuey*] | [Valuez = *valuez*])

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
valuex	real	0
valuey	real	0
valuez	real	0

Constant_Traction

Syntax

Bc Lumped_Flux For Mesh ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant_Traction ([Toggle = *toggle*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Constant traction mesh flux

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
x	real	0
y	real	0
z	real	0

Damping

Syntax

Bc Lumped_Flux For Mesh ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Damping ([Toggle = *toggle*] | [Scale_Factor = *scale_factor*])

Summary

Dampen surface displacements.

Description

$$F = \frac{-2\mu s}{h}\frac{\partial u}{\partial t}$$

where 2μ is the Lame constant, s the optional "scale_factor" parameter, h the mesh size, and u the mesh displacements. This BC is primarily intended for use in conjunction with a rotated BC that enforces the normal mesh displacement, and serves to dampen tangential mesh displacements on the surface.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
scale_factor	real	1

Electric_Traction

Syntax

Bc Lumped_Flux For Mesh ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Electric_Traction ([Toggle = *toggle*] | [Sign = *sign*])

Summary

Electric traction mesh flux

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
sign	real	1

Penalized_Normal_Displacement

Syntax

Bc Lumped_Flux For Mesh ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Penalized_Normal_Displacement ([Toggle = *toggle*] | [Stiffness_Factor = *stiffness_factor*])

Summary

Penalized normal flux

Description

Enforces a weak zero-normal-displacement condition by applying a mesh flux of

$$\vec{F} = -2\mu S_f \left(\vec{D} \cdot \vec{n}\right) \vec{n}$$

Where 2μ is the mesh `TWO_MU` value, and S_f is the stiffness factor.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
stiffness_factor	real	1e+06

Pressure

Syntax

Bc Lumped_Flux For Mesh ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Pressure ([Toggle = *toggle*] | [User_Field_Name = *user_field_name*] | [Use_Bulk_Node = *use_bulk_node*] | [P = *p*] | [C_T = *c_t*] | [C_X = *c_x*] | [C_Y = *c_y*] | [C_Z = *c_z*] | [C_Cos_W = *c_cos_w*] | [C_Sin_W = *c_sin_w*] | [C1_Cos = *c1_cos*] | [C1_Sin = *c1_sin*])

Summary

Pressure mesh flux

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
user_field_name	"string"	–
use_bulk_node	"string"	–
p	real	0
c_t	real	0
c_x	real	0
c_y	real	0
c_z	real	0
c_cos_w	real	0
c_sin_w	real	0
c1_cos	real	0
c1_sin	real	0

Pressure_User_Function

Syntax

Bc Lumped_Flux For Mesh ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Pressure_User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Pressure user function mesh flux

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1

Shear_Free

Syntax

Bc Lumped_Flux For Mesh ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Shear_Free [Toggle = *toggle*]

Summary

Shear-free traction mesh flux

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–

Specified_Normal_Velocity

Syntax

Bc Lumped_Flux For Mesh ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Specified_Normal_Velocity ([Toggle = *toggle*] | [Stiffness_Factor = *stiffness_factor*])

Summary

Penalized normal mesh velocity

Description

Enforces a weak specified normal velocity condition by applying a mesh flux of

$$\vec{F} = 2\mu S_f \left(V_m - \dot{\vec{D}} \cdot \vec{n} \right) \vec{n}$$

Where 2μ is the mesh `TWO_MU` value, and S_f is the stiffness factor. The normal mesh velocity, V_m is provided by the `NORMAL_MESH_VELOCITY` expression on the surface (scalar, positive for outward mesh velocity).

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
stiffness_factor	real	1e+06

Transient_Traction

Syntax

Bc Lumped_Flux For Mesh ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Transient_Traction ([Toggle = *toggle*] | [A_X = *a_x*] | [A_Y = *a_y*] | [A_Z = *a_z*] | [B_X = *b_x*] | [B_Y = *b_y*] | [B_Z = *b_z*])

Summary

Transient traction mesh flux

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
a_x	real	0
a_y	real	0
a_z	real	0
b_x	real	0
b_y	real	0
b_z	real	0

Vector_String_Function

Syntax

Bc Lumped_Flux For Mesh ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Vector_String_Function ([Toggle = *toggle*] | F_X = *f_x* | F_Y = *f_y* | [F_Z = *f_z*])

Summary

Value from a string function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f_x	"string"	–
f_y	"string"	–
f_z	"string"	–

Mesh_X

Constant

Syntax

Bc Lumped_Flux For Mesh_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | Value = *value*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
value	real	–

Exponential

Syntax

Bc Lumped_Flux For Mesh_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Lumped_Flux For Mesh_X ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Bc Lumped_Flux For Mesh_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc Lumped_Flux For Mesh_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Lumped_Flux For Mesh_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f	"string"	–

User_Field

Syntax

Bc Lumped_Flux For Mesh_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc Lumped_Flux For Mesh_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc Lumped_Flux For Mesh_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

Mesh_Y

Constant

Syntax

Bc Lumped_Flux For Mesh_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | Value = *value*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
value	real	–

Exponential

Syntax

Bc Lumped_Flux For Mesh_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Lumped_Flux For Mesh_Y ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Bc Lumped_Flux For Mesh_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc Lumped_Flux For Mesh_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Lumped_Flux For Mesh_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f	"string"	–

User_Field

Syntax

Bc Lumped_Flux For Mesh_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc Lumped_Flux For Mesh_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc Lumped_Flux For Mesh_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

Mesh_Z

Constant

Syntax

Bc Lumped_Flux For Mesh_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | Value = *value*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
value	real	–

Exponential

Syntax

Bc Lumped_Flux For Mesh_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Lumped_Flux For Mesh_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
global_name	“string”	–
component	integer	0

Nd_Table

Syntax

Bc Lumped_Flux For Mesh_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc Lumped_Flux For Mesh_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Lumped_Flux For Mesh_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f	"string"	–

User_Field

Syntax

Bc Lumped_Flux For Mesh_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
scaling	real	1
global_var	“string”	–
time_function	“string”	–

User_Function

Syntax

Bc Lumped_Flux For Mesh_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc Lumped_Flux For Mesh_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

Mixture_Fraction

Constant

Syntax

Bc Lumped_Flux For Mixture_Fraction ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | *{flux | value}* = *flux*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
flux	real	–

Exponential

Syntax

Bc Lumped_Flux For Mixture_Fraction ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Lumped_Flux For Mixture_Fraction ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Bc Lumped_Flux For Mixture_Fraction ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Open_Flow_Convection

Syntax

Bc Lumped_Flux For Mixture_Fraction ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Open_Flow_Convection ([Toggle = *toggle*] | Reference_Value = *Reference_Value*)

Summary

Open flow convection BC for the mixture fraction equation

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Reference_Value	real	–

Polynomial

Syntax

Bc Lumped_Flux For Mixture_Fraction ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Lumped_Flux For Mixture_Fraction ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f	"string"	–

User_Field

Syntax

Bc Lumped_Flux For Mixture_Fraction ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc Lumped_Flux For Mixture_Fraction ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc Lumped_Flux For Mixture_Fraction ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

Momentum

Capillary

Syntax

Bc Lumped_Flux For Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Capillary [Toggle = *toggle*]

Summary

LS capillary momentum flux

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–

Capillary_Gap

Syntax

Bc Lumped_Flux For Momentum ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Capillary_Gap ([Toggle = *toggle*] | V_W = *v_w* | G = *g* | Theta = *theta*)

Summary

This model has been deprecated

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
v_w	real	–
g	real	–
theta	real	–

Constant

Syntax

Bc Lumped_Flux For Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
x	real	0
y	real	0
z	real	0

Constant_Traction

Syntax

Bc Lumped_Flux For Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant_Traction ([Toggle = *toggle*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
x	real	0
y	real	0
z	real	0

Darcy_Slip

Syntax

Bc Lumped_Flux For Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Darcy_Slip ([Toggle = *toggle*] | [Vs_X = *vs_x*] | [Vs_Y = *vs_y*] | [Vs_Z = *vs_z*] | K = *k*)

Summary

Darcy slip momentum flux

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
vs_x	real	0
vs_y	real	0
vs_z	real	0
k	real	–

Electric_Traction

Syntax

Bc Lumped_Flux For Momentum ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Electric_Traction ([Toggle = *toggle*] | [Sign = *sign*])

Summary

Electric traction momentum flux

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
sign	real	1

Equilibrium_Capillary_Gap

Syntax

Bc Lumped_Flux For Momentum ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Equilibrium_Capillary_Gap ([Toggle = *toggle*] | Theta = *theta*)

Summary

This model has been deprecated

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
theta	real	–

Exponential_Vapor_Recoil_Pressure

Syntax

Bc Lumped_Flux For Momentum ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential_Vapor_Recoil_Pressure ([Toggle = *toggle*] | Tboil = *tboil* | L = *l* | Pa = *pa* | [Beta = *beta*] | Mw = *mw* | [Tl = *tl*] | [Th = *th*])

Summary

Exponential vapor recoil pressure momentum flux

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
tboil	real	–
l	real	–
pa	real	–
beta	real	1
mw	real	–
tl	real	-1
th	real	-1

Fixed_Curvature

Syntax

Bc Lumped_Flux For Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Fixed_Curvature ([Toggle = *toggle*] | Curvature = *curvature*)

Summary

This model has been deprecated and removed from the code.

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
curvature	real	–

Flow_Hydrostatic

Syntax

Bc Lumped_Flux For Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Flow_Hydrostatic ([Toggle = *toggle*] | [P_Ref = *p_ref*] | [Gx = *gx*] | [Gy = *gy*] | [Gz = *gz*])

Summary

Flow hydrostatic momentum flux

Description

The following options are present in most boundary conditions/sources. The

meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
p_ref	real	0
gx	real	0
gy	real	0
gz	real	0

Free_Open_Flow

Syntax

Bc Lumped_Flux For Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Free_Open_Flow ([Toggle = *toggle*] | Pressure = *pressure*)

Summary

Open flow for momentum; apply the full stress

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
pressure	real	–

Lens_Particles_Pressure

Syntax

Bc Lumped_Flux For Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Lens_Particles_Pressure ([Toggle = *toggle*] | Src_X = *src_x* | Dir_X = *dir_x* | Vel_X = *vel_x* | Src_Y = *src_y* | Dir_Y = *dir_y* | Vel_Y = *vel_y* | [Src_Z = *src_z*] | [Dir_Z = *dir_z*] | [Vel_Z = *vel_z*] | Massflowrate = *massFlowRate* | T_Liq = *T_liq* | T_Solid = *T_solid* | [T_On = *t_on*] | [T_Off = *t_off*] | Particlespeed = *particleSpeed* | Radius = *radius*)

Summary

LENS particles pressure momentum flux

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
src_x	real	–
dir_x	real	–
vel_x	real	–
src_y	real	–
dir_y	real	–
vel_y	real	–
src_z	real	0
dir_z	real	0
vel_z	real	0
massFlowRate	real	–
T_liq	real	–
T_solid	real	–
t_on	real	0
t_off	real	1.798e+308
particleSpeed	real	–
radius	real	–

Ls_Capillary

Syntax

Bc Lumped_Flux For Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Ls_Capillary [Toggle = *toggle*]

Summary

LS capillary momentum flux

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–

Ls_Oriented_Slip

Syntax

Bc Lumped_Flux For Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Ls_Oriented_Slip ([Toggle = *toggle*] | [Vs_X = *vs_x*] | [Vs_Y = *vs_y*] | [Vs_Z = *vs_z*] | [Beta_Normal_A = *beta_normal_a*] | [Beta_Tangent_A = *beta_tangent_a*] | [Beta_Normal_B = *beta_normal_b*] | [Beta_Tangent_B = *beta_tangent_b*])

Summary

Oriented slip momentum flux

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
vs_x	real	0
vs_y	real	0
vs_z	real	0
beta_normal_a	real	0
beta_tangent_a	real	0
beta_normal_b	real	0
beta_tangent_b	real	0

Node_Normal_Capillary

Syntax

Bc Lumped_Flux For Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Node_Normal_Capillary [Toggle = *toggle*]

Summary

This model has been deprecated and removed from the code.

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–

Non_Ibp_Pressure

Syntax

Bc Lumped_Flux For Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Non_Ibp_Pressure ([Toggle = *toggle*] | [Pressure = *pressure*])

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
pressure	"string"	–

Open_Flow

Syntax

Bc Lumped_Flux For Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Open_Flow ([Toggle = *toggle*] | [Pressure = *pressure*] | [Total_Pressure = *total_pressure*])

Summary

Open flow for momentum; enforce normal gradient to be zero

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
pressure	real	0
total_pressure	real	0

Oriented_Slip

Syntax

Bc Lumped_Flux For Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Oriented_Slip ([Toggle = *toggle*] | [Vs_X = *vs_x*] | [Vs_Y = *vs_y*] | [Vs_Z = *vs_z*] | [Vs_Expr = *vs_expr*] | Beta_Normal = *beta_normal* | Beta_Tangent = *beta_tangent*)

Summary

Oriented slip momentum flux. Essentially a sum of the Navier-Slip condition

and a penalty condition for enforcing velocity in the normal direction.

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
vs_x	real	0
vs_y	real	0
vs_z	real	0
vs_expr	"string"	–
beta_normal	real	–
beta_tangent	real	–

Pressure

Syntax

Bc Lumped_Flux For Momentum ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Pressure ([Toggle = *toggle*] | [User_Field_Name = *user_field_name*] | [Use_Bulk_Node = *use_bulk_node*] | [P = *p*] | [C_T = *c_t*] | [C_X = *c_x*] | [C_Y = *c_y*] | [C_Z = *c_z*] | [C_Cos_W = *c_cos_w*] | [C_Sin_W = *c_sin_w*] | [C1_Cos = *c1_cos*] | [C1_Sin = *c1_sin*])

Summary

Pressure momentum flux

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
user_field_name	"string"	–
use_bulk_node	"string"	–
p	real	0
c_t	real	0
c_x	real	0
c_y	real	0
c_z	real	0
c_cos_w	real	0
c_sin_w	real	0
c1_cos	real	0
c1_sin	real	0

Pressure_User_Function

Syntax

Bc Lumped_Flux For Momentum ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Pressure_User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Pressure user function momentum flux

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is

active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1

Projected_Capillary

Syntax

Bc Lumped_Flux For Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Projected_Capillary [Toggle = *toggle*]

Summary

Projected capillary momentum flux

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–

Reconstructed_Curvature

Syntax

Bc Lumped_Flux For Momentum ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Reconstructed_Curvature ([Toggle = *toggle*] | [Ls_Size = *ls_size*] | [Fixed_Curvature = *fixed_curvature*])

Summary

This model has been deprecated and removed from the code.

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
ls_size	integer	0
fixed_curvature	real	-1

Simple_Spring_Force

Syntax

Bc Lumped_Flux For Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Simple_Spring_Force ([Toggle = *toggle*] | [Kx = *kx*] | [Ky = *ky*] | [Kz = *kz*] | [X0 = *x0*] | [Y0 = *y0*] | [Z0 = *z0*])

Summary

This model has been deprecated and removed from the code.

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
kx	real	0
ky	real	0
kz	real	0
x0	real	0
y0	real	0
z0	real	0

Slip

Syntax

Bc Lumped_Flux For Momentum ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Slip ([Toggle = *toggle*] | [Vs_X = *vs_x*] | [Vs_Y = *vs_y*] | [Vs_Z = *vs_z*] | Beta = *beta*)

Summary

Slip momentum flux

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
vs_x	real	0
vs_y	real	0
vs_z	real	0
beta	real	–

Slip_Length

Syntax

Bc Lumped_Flux For Momentum ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Slip_Length ([Toggle = *toggle*] | [Vs_X = *vs_x*] | [Vs_Y = *vs_y*] | [Vs_Z = *vs_z*] | Beta = *beta*)

Summary

Slip length momentum flux

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
vs_x	real	0
vs_y	real	0
vs_z	real	0
beta	real	–

Spring_Force

Syntax

Bc Lumped_Flux For Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Spring_Force ([Toggle = *toggle*] | Mult = *Mult*)

Summary

Spring force for momentum boundary condition

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Mult	real	–

Symmetry_Flow

Syntax

Bc Lumped_Flux For Momentum ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Symmetry_Flow [Toggle = *toggle*]

Summary

Symmetry flow momentum flux

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–

Transient_Traction

Syntax

Bc Lumped_Flux For Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Transient_Traction ([Toggle = *toggle*] | [A_X = *a_x*] | [A_Y = *a_y*] | [A_Z = *a_z*] | [B_X = *b_x*] | [B_Y = *b_y*] | [B_Z = *b_z*])

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
a_x	real	0
a_y	real	0
a_z	real	0
b_x	real	0
b_y	real	0
b_z	real	0

Vapor_Recoil_Pressure

Syntax

Bc Lumped_Flux For Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Vapor_Recoil_Pressure ([Toggle = *toggle*] | Tboil = *tboil*)

Summary

Vapor recoil pressure momentum flux

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
tboil	real	–

Vector_String_Function

Syntax

Bc Lumped_Flux For Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Vector_String_Function ([Toggle = *toggle*] | F_X = *f_x* | F_Y = *f_y* | [F_Z = *f_z*])

Summary

Value from a string function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
f_x	“string”	–
f_y	“string”	–
f_z	“string”	–

Momentum_X

Constant

Syntax

Bc Lumped_Flux For Momentum_X ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | Value = *value*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
value	real	–

Exponential

Syntax

Bc Lumped_Flux For Momentum_X ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Lumped_Flux For Momentum_X ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Bc Lumped_Flux For Momentum_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc Lumped_Flux For Momentum_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Lumped_Flux For Momentum_X ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f	"string"	–

User_Field

Syntax

Bc Lumped_Flux For Momentum_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
scaling	real	1
global_var	“string”	–
time_function	“string”	–

User_Function

Syntax

Bc Lumped_Flux For Momentum_X ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc Lumped_Flux For Momentum_X ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

Momentum_Y

Constant

Syntax

Bc Lumped_Flux For Momentum_Y ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | Value = *value*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
value	real	–

Exponential

Syntax

Bc Lumped_Flux For Momentum_Y ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Lumped_Flux For Momentum_Y ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Bc Lumped_Flux For Momentum_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc Lumped_Flux For Momentum_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Lumped_Flux For Momentum_Y ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f	"string"	–

User_Field

Syntax

Bc Lumped_Flux For Momentum_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc Lumped_Flux For Momentum_Y ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc Lumped_Flux For Momentum_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

Momentum_Z

Constant

Syntax

Bc Lumped_Flux For Momentum_Z ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | Value = *value*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
value	real	–

Exponential

Syntax

Bc Lumped_Flux For Momentum_Z ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Lumped_Flux For Momentum_Z ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Bc Lumped_Flux For Momentum_Z ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc Lumped_Flux For Momentum_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Lumped_Flux For Momentum_Z ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f	"string"	–

User_Field

Syntax

Bc Lumped_Flux For Momentum_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc Lumped_Flux For Momentum_Z ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc Lumped_Flux For Momentum_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

Porous_Enthalpy

Constant

Syntax

Bc Lumped_Flux For Porous_Enthalpy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | Value = *value*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
value	real	–

Convective_Outflow

Syntax

Bc Lumped_Flux For Porous_Enthalpy ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Convective_Outflow ([Toggle = *toggle*] | [Ref_H = *ref_h*])

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
ref_h	real	0

Enclosure_Radiation

Syntax

Bc Lumped_Flux For Porous_Enthalpy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Enclosure_Radiation ([Toggle = *toggle*] | [Multiplier = *multiplier*] | Enclosure = *enclosure*)

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
multiplier	real	1
enclosure	“string”	–

Exponential

Syntax

Bc Lumped_Flux For Porous_Enthalpy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the

constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Fluid_Robin_Coupled

Syntax

Bc Lumped_Flux For Porous_Enthalpy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Fluid_Robin_Coupled ([Toggle = *toggle*] | Enthalpy_Field = *enthalpy_field* | Enthalpy_Diffusive_Flux_Field = *enthalpy_diffusive_flux_field* | Coeff_Field = *coeff_field*)

Summary

Boundary condition for loose coupling of enthalpy to a free fluid region

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
enthalpy_field	"string"	–
enthalpy_diffusive_flux_field	"string"	–
coeff_field	"string"	–

Fluid_Robin_Coupled_One_Region

Syntax

Bc Lumped_Flux For Porous_Enthalpy ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Fluid_Robin_Coupled_One_Region [Toggle = *toggle*]

Summary

Boundary condition for loose coupling of enthalpy to a free fluid region

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–

Fluid_Robin_Coupled_With_Solid_Convection

Syntax

Bc Lumped_Flux For Porous_Enthalpy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Fluid_Robin_Coupled_With_Solid_Convection ([Toggle = *toggle*] | Enthalpy_Field = *enthalpy_field* | Enthalpy_Diffusive_Flux_Field = *enthalpy_diffusive_flux_field* | Coeff_Field = *coeff_field*)

Summary

Boundary condition for loose coupling of enthalpy to a free fluid region

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
enthalpy_field	“string”	–
enthalpy_diffusive_flux_field	“string”	–
coeff_field	“string”	–

Fluid_Robin_Coupled_With_Solid_Convection_One_Region

Syntax

Bc Lumped_Flux For Porous_Enthalpy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Fluid_Robin_Coupled_With_Solid_Convection_One_Region [Toggle = *toggle*]

Summary

Boundary condition for loose coupling of enthalpy to a free fluid region

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–

Fluid_Solid_Convection_Coupled

Syntax

Bc Lumped_Flux For Porous_Enthalpy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Fluid_Solid_Convection_Coupled ([Toggle = *toggle*] | [Multiplier = *multiplier*] | Fluid_Temperature_Field = *fluid_temperature_field*)

Summary

Boundary condition for loose coupling of enthalpy to a free fluid region

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
multiplier	real	1
fluid_temperature_field	"string"	–

Fluid_Solid_Convection_Coupled_One_Region

Syntax

Bc Lumped_Flux For Porous_Enthalpy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Fluid_Solid_Convection_Coupled_One_Region ([Toggle = *toggle*] | [Multiplier = *multiplier*])

Summary

Boundary condition for loose coupling of enthalpy to a free fluid region

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
multiplier	real	1

Generalized_Nat_Conv

Syntax

Bc Lumped_Flux For Porous_Enthalpy ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Generalized_Nat_Conv ([Toggle = *toggle*] | [Multiplier = *multiplier*])

Summary

Generalized natural convection flux BC

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
multiplier	real	1

Generalized_Rad

Syntax

Bc Lumped_Flux For Porous_Enthalpy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Generalized_Rad ([Toggle = *toggle*] | [Multiplier = *multiplier*] | [T_Ref = *t_ref*])

Summary

Radiative energy flux, with values for emissivity, form factor and irradiation taken from expressions

Description

Flux is:

$$M\epsilon f\left(\sigma T^4 - I\right)$$

M is MULTIPLIER. σ is the Stefan-Boltzmann constant. ϵ and F are expressions for emissivity and form factor. T is surface temperature. I is the expression for irradiation. If I does not already exist, it will be constructed as σT_{ref}^4, where T_{ref} is either provided by the BC Rad Reference Temperature expression or the inline T_{ref} parameter (constant or string function).

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
multiplier	real	1
t_ref	“string”	–

Global

Syntax

Bc Lumped_Flux For Porous_Enthalpy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
global_name	“string”	–
component	integer	0

Material_Phase_Sum_Open

Syntax

Bc Lumped_Flux For Porous_Enthalpy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Material_Phase_Sum_Open ([Toggle = *toggle*] | [Pressure = *pressure*] | [Entrained_Value = *entrained_value*])

Summary

Open boundary condition for a single temperature model, wherein the phases are assumed to be in thermal equilibrium. The advective flux of enthalpy for

each phase is summed at the open.Must provide the BOUNDARY_ENTRAINED_ENTHALPY expression for entrainment.

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
pressure	real	0
entrained_value	real	0

Mesh_Motion

Syntax

Bc Lumped_Flux For Porous_Enthalpy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Mesh_Motion [Toggle = *toggle*]

Summary

Moving boundary condition for Arbitrary Lagrangian-Eulerian (ALE) mesh motion in relative flux form. Accounts for the boundary advective flux due to general mesh motion.

Description

Applies the following boundary flux

$$-\int q\hat{\mathbf{u}}\cdot\mathbf{n}dS,$$

where $\hat{u}$ is the mesh velocity and where the integrated scalar q depends on the equation. For the gas phase `porous_enthalpy` equation with a `transported`

`enthalpy = porous`, $q = \phi\rho_g h_g$. Similarly, for the gas phase `mass_balance` equation, $q = \phi\rho_g$ is the bulk mass density.

This flux boundary condition is automatically applied to the mass balance and porous enthalpy equations when the `use relative mass flux` porous flow option is turned on.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle
If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–

Moving_Boundary

Syntax

Bc Lumped_Flux For Porous_Enthalpy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Moving_Boundary [Toggle = *toggle*]

Summary

No penetration moving boundary condition, valid for Lagrangian mesh motion in non-relative flux form or relative flux form.

Description

Applies the following boundary flux

$$\int q\hat{\mathbf{u}} \cdot \mathbf{n} dS,$$

where $\hat{u}$ is the mesh velocity and where the integrated scalar q depends on the equation. If we consider the gas phase `porous_enthalpy` equation with a `transported enthalpy = porous`, then $q = \phi\rho_g h_g$. Similarly, for the gas phase `mass_balance` equation, $q = \phi\rho_g$ is the bulk mass density.

If we consider the gas phase `mass_balance` equation, with a gas phase porous velocity relative to the moving solid phase, then physically this boundary condition is derived from applying a no flux condition at moving boundary i.e.,

$$\mathbf{u}_g \cdot \mathbf{n} = \phi \hat{\mathbf{u}} \cdot \mathbf{n}.$$

In the Lagrangian setting, this flux is required to impose a no flux condition at the moving boundary with both the non-relative form (default) and relative form of the advection term. When using the relative form, typically a do-nothing condition would be sufficient to impose no flux; however, to simplify general ALE mesh motion for porous flow equations (i.e., mass balance and porous enthalpy) we automatically provide the mesh motion flux on boundaries for blocks that have mesh motion. Consequently, to impose a zero flux, this term is needed for cancellation.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–

Nd_Table

Syntax

Bc Lumped_Flux For Porous_Enthalpy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

One_Temp_Multiphase_Open

Syntax

Bc Lumped_Flux For Porous_Enthalpy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = One_Temp_Multiphase_Open ([Toggle = *toggle*] | [Pressure = *pressure*] | [Entrained_Value = *entrained_value*])

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
pressure	real	0
entrained_value	real	0

Open

Syntax

Bc Lumped_Flux For Porous_Enthalpy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Open ([Toggle = *toggle*] | [Pressure = *pressure*] | [Entrained_Value = *entrained_value*])

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
pressure	real	0
entrained_value	real	0

Polynomial

Syntax

Bc Lumped_Flux For Porous_Enthalpy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
variable	“string”	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Lumped_Flux For Porous_Enthalpy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
f	“string”	–

User_Field

Syntax

Bc Lumped_Flux For Porous_Enthalpy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc Lumped_Flux For Porous_Enthalpy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc Lumped_Flux For Porous_Enthalpy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

User_Vector_Field

Syntax

Bc Lumped_Flux For Porous_Enthalpy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Vector_Field ([Toggle = *toggle*] | Name = *name* | [Multiplier = *multiplier*])

Summary

User vector field enthalpy flux

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
multiplier	real	1

Porous_Species

Constant

Syntax

Bc Lumped_Flux For Porous_Species ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | *{flux | value}* = *flux*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
flux	real	–

Exponential

Syntax

Bc Lumped_Flux For Porous_Species ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Lumped_Flux For Porous_Species ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
global_name	"string"	–
component	integer	0

Moving_Boundary

Syntax

Bc Lumped_Flux For Porous_Species ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Moving_Boundary [Toggle = *toggle*]

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–

Nd_Table

Syntax

Bc Lumped_Flux For Porous_Species ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Open_Flow

Syntax

Bc Lumped_Flux For Porous_Species ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Open_Flow [Toggle = *toggle*]

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–

Polynomial

Syntax

Bc Lumped_Flux For Porous_Species ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Lumped_Flux For Porous_Species ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name.

Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f	"string"	–

User_Field

Syntax

Bc Lumped_Flux For Porous_Species ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc Lumped_Flux For Porous_Species ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc Lumped_Flux For Porous_Species ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

Solid

Constant

Syntax

Bc Lumped_Flux For Solid ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | [Valuex = *valuex*] | [Valuey = *valuey*] | [Valuez = *valuez*])

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
valuex	real	0
valuey	real	0
valuez	real	0

Constant_Traction

Syntax

Bc Lumped_Flux For Solid ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant_Traction ([Toggle = *toggle*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Constant traction solid flux

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
x	real	0
y	real	0
z	real	0

Electric_Traction

Syntax

Bc Lumped_Flux For Solid ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Electric_Traction ([Toggle = *toggle*] | [Sign = *sign*])

Summary

Electric traction solid flux

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
sign	real	1

Pressure

Syntax

Bc Lumped_Flux For Solid ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Pressure ([Toggle = *toggle*] | [User_Field_Name = *user_field_name*] | [Use_Bulk_Node = *use_bulk_node*] | [P = *p*] | [C_T = *c_t*] | [C_X = *c_x*] | [C_Y = *c_y*] | [C_Z = *c_z*] | [C_Cos_W = *c_cos_w*] | [C_Sin_W = *c_sin_w*] | [C1_Cos = *c1_cos*] | [C1_Sin = *c1_sin*])

Summary

Pressure solid flux

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
user_field_name	“string”	–
use_bulk_node	“string”	–
p	real	0
c_t	real	0
c_x	real	0
c_y	real	0
c_z	real	0
c_cos_w	real	0
c_sin_w	real	0
c1_cos	real	0
c1_sin	real	0

Pressure_User_Function

Syntax

Bc Lumped_Flux For Solid ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Pressure_User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Pressure user function solid flux

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1

Shear_Free

Syntax

Bc Lumped_Flux For Solid ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Shear_Free [Toggle = *toggle*]

Summary

Shear-free traction solid flux

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–

Spring_Force

Syntax

Bc Lumped_Flux For Solid ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Spring_Force ([Toggle = *toggle*] | Mult = *Mult*)

Summary

Spring force solid flux

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Mult	real	–

Transient_Traction

Syntax

Bc Lumped_Flux For Solid ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Transient_Traction ([Toggle = *toggle*] | [A_X = *a_x*] | [A_Y = *a_y*] | [A_Z = *a_z*] | [B_X = *b_x*] | [B_Y = *b_y*] | [B_Z = *b_z*])

Summary

Transient traction solid flux

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
a_x	real	0
a_y	real	0
a_z	real	0
b_x	real	0
b_y	real	0
b_z	real	0

Vector_String_Function

Syntax

Bc Lumped_Flux For Solid ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Vector_String_Function ([Toggle = *toggle*] | F_X = *f_x* | F_Y = *f_y* | [F_Z = *f_z*])

Summary

Value from a string function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
f_x	“string”	–
f_y	“string”	–
f_z	“string”	–

Solid_X

Constant

Syntax

Bc Lumped_Flux For Solid_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | Value = *value*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
value	real	–

Exponential

Syntax

Bc Lumped_Flux For Solid_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Lumped_Flux For Solid_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Bc Lumped_Flux For Solid_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc Lumped_Flux For Solid_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Lumped_Flux For Solid_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f	"string"	–

User_Field

Syntax

Bc Lumped_Flux For Solid_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc Lumped_Flux For Solid_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc Lumped_Flux For Solid_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

Solid_Y

Constant

Syntax

Bc Lumped_Flux For Solid_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | Value = *value*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
value	real	–

Exponential

Syntax

Bc Lumped_Flux For Solid_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Lumped_Flux For Solid_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
global_name	“string”	–
component	integer	0

Nd_Table

Syntax

Bc Lumped_Flux For Solid_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc Lumped_Flux For Solid_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Lumped_Flux For Solid_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f	"string"	–

User_Field

Syntax

Bc Lumped_Flux For Solid_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc Lumped_Flux For Solid_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc Lumped_Flux For Solid_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

Solid_Z

Constant

Syntax

Bc Lumped_Flux For Solid_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | Value = *value*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
value	real	–

Exponential

Syntax

Bc Lumped_Flux For Solid_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Lumped_Flux For Solid_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Bc Lumped_Flux For Solid_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc Lumped_Flux For Solid_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Lumped_Flux For Solid_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f	"string"	–

User_Field

Syntax

Bc Lumped_Flux For Solid_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc Lumped_Flux For Solid_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc Lumped_Flux For Solid_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

Species

Constant

Syntax

Bc Lumped_Flux For Species ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | *{flux | value}* = *flux*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
flux	real	–

Exponential

Syntax

Bc Lumped_Flux For Species ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Free_Open_Flow

Syntax

Bc Lumped_Flux For Species ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Free_Open_Flow [Toggle = *toggle*]

Summary

This Flux BC sets the surface normal species flux equal to the diffusive flux.

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–

Global

Syntax

Bc Lumped_Flux For Species ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
global_name	“string”	–
component	integer	0

Interface

Syntax

Bc Lumped_Flux For Species ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Interface [Toggle = *toggle*]

Summary

Flux BC model of species for moving interfaces

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–

Nat_Conv

Syntax

Bc Lumped_Flux For Species ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nat_Conv ([Toggle = *toggle*] | K = *k* | Yinf = *yinf*)

Summary

natconv species flux

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
k	real	–
yinf	real	–

Nd_Table

Syntax

Bc Lumped_Flux For Species ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Open_Adv_Flow

Syntax

Bc Lumped_Flux For Species ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Open_Adv_Flow ([Toggle = *toggle*] | [Far_Field_Entrainment_Value = *far_field_entrainment_value*])

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
far_field_entrainment_value	real	0

Polynomial

Syntax

Bc Lumped_Flux For Species ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Lumped_Flux For Species ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f	"string"	–

User_Field

Syntax

Bc Lumped_Flux For Species ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc Lumped_Flux For Species ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc Lumped_Flux For Species ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

Voltage

Constant

Syntax

Bc Lumped_Flux For Voltage ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | *{flux | value}* = *flux*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
flux	real	–

Exponential

Syntax

Bc Lumped_Flux For Voltage ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Lumped_Flux For Voltage ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Bc Lumped_Flux For Voltage ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc Lumped_Flux For Voltage ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Lumped_Flux For Voltage ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f	"string"	–

User_Field

Syntax

Bc Lumped_Flux For Voltage ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc Lumped_Flux For Voltage ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc Lumped_Flux For Voltage ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

Bc No_Coverage_Flux

Ampere

Constant

Syntax

Bc No_Coverage_Flux For Ampere ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | [Valuex = *valuex*] | [Valuey = *valuey*] | [Valuez = *valuez*])

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
valuex	real	0
valuey	real	0
valuez	real	0

Vector_String_Function

Syntax

Bc No_Coverage_Flux For Ampere ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Vector_String_Function ([Toggle = *toggle*] | F_X = *f_x* | F_Y = *f_y* | [F_Z = *f_z*])

Summary

Value from a string function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f_x	"string"	–
f_y	"string"	–
f_z	"string"	–

Ampere_X

Constant

Syntax

Bc No_Coverage_Flux For Ampere_X ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | Value = *value*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
value	real	–

Exponential

Syntax

Bc No_Coverage_Flux For Ampere_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc No_Coverage_Flux For Ampere_X ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Bc No_Coverage_Flux For Ampere_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc No_Coverage_Flux For Ampere_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc No_Coverage_Flux For Ampere_X ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f	"string"	–

User_Field

Syntax

Bc No_Coverage_Flux For Ampere_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc No_Coverage_Flux For Ampere_X ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc No_Coverage_Flux For Ampere_X ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

Ampere_Y

Constant

Syntax

Bc No_Coverage_Flux For Ampere_Y ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | Value = *value*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
value	real	–

Exponential

Syntax

Bc No_Coverage_Flux For Ampere_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc No_Coverage_Flux For Ampere_Y ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Bc No_Coverage_Flux For Ampere_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc No_Coverage_Flux For Ampere_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc No_Coverage_Flux For Ampere_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f	"string"	–

User_Field

Syntax

Bc No_Coverage_Flux For Ampere_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc No_Coverage_Flux For Ampere_Y ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc No_Coverage_Flux For Ampere_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

Ampere_Z

Constant

Syntax

Bc No_Coverage_Flux For Ampere_Z ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | Value = *value*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
value	real	–

Exponential

Syntax

Bc No_Coverage_Flux For Ampere_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc No_Coverage_Flux For Ampere_Z ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Bc No_Coverage_Flux For Ampere_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc No_Coverage_Flux For Ampere_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc No_Coverage_Flux For Ampere_Z ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f	"string"	–

User_Field

Syntax

Bc No_Coverage_Flux For Ampere_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc No_Coverage_Flux For Ampere_Z ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc No_Coverage_Flux For Ampere_Z ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

Charge_Density

Constant

Syntax

Bc No_Coverage_Flux For Charge_Density ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | *{flux | value}* = *flux*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
flux	real	–

Exponential

Syntax

Bc No_Coverage_Flux For Charge_Density ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc No_Coverage_Flux For Charge_Density ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
global_name	"string"	–
component	integer	0

Nat_E

Syntax

Bc No_Coverage_Flux For Charge_Density ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nat_E ([Toggle = *toggle*] | [Sigma = *sigma*])

Summary

This model is now deprecated

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
sigma	"string"	–

Nd_Table

Syntax

Bc No_Coverage_Flux For Charge_Density ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc No_Coverage_Flux For Charge_Density ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
variable	“string”	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc No_Coverage_Flux For Charge_Density ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
f	“string”	–

User_Field

Syntax

Bc No_Coverage_Flux For Charge_Density ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc No_Coverage_Flux For Charge_Density ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc No_Coverage_Flux For Charge_Density ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

Continuity

Constant

Syntax

Bc No_Coverage_Flux For Continuity ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | Value = *value*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
value	real	–

Exponential

Syntax

Bc No_Coverage_Flux For Continuity ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Extrapolated_Pressure

Syntax

Bc No_Coverage_Flux For Continuity ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Extrapolated_Pressure [Toggle = *toggle*]

Summary

This model has been deprecated and removed from the code.

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–

Global

Syntax

Bc No_Coverage_Flux For Continuity ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Bc No_Coverage_Flux For Continuity ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc No_Coverage_Flux For Continuity ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
variable	“string”	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Pspg

Syntax

Bc No_Coverage_Flux For Continuity ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Pspg ([Toggle = *toggle*] | [Tausurfacefactor = *TauSurfaceFactor*] | [Tausurfacejacobianfactor = *TauSurfaceJacobianFactor*])

Summary

Continuity PSPG flux BC

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
TauSurfaceFactor	real	1
TauSurfaceJacobianFactor	real	1

Scalar_String_Function

Syntax

Bc No_Coverage_Flux For Continuity ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f	"string"	–

User_Field

Syntax

Bc No_Coverage_Flux For Continuity ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc No_Coverage_Flux For Continuity ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc No_Coverage_Flux For Continuity ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

Current

Calore_User_Sub

Syntax

Bc No_Coverage_Flux For Current ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Calore_User_Sub ([Toggle = *toggle*] | Name = *name* | Type = *type* | [Multiplier = *multiplier*] | [Material_Data_Block = *material_data_block*] | [Data = *data*] | [Scaling_Field = *scaling_field*])

Summary

Values from a Calore user subroutine

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
type	“string”	–
multiplier	real	1
material_data_block	“string”	–
data	“string”	–
scaling_field	“string”	–

Constant

Syntax

Bc No_Coverage_Flux For Current ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | *{flux | value}* = *flux*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
flux	real	–

Constant_With_Cutoff_Voltage

Syntax

Bc No_Coverage_Flux For Current ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant_With_Cutoff_Voltage ([Toggle = *toggle*] | Value = *value* | Cutoff_Value = *cutoff_value*)

Summary

Apply a constant current until a cutoff voltage is crossed anywhere on the surface. After that apply 0 current.

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
value	real	–
cutoff_value	real	–

Exponential

Syntax

Bc No_Coverage_Flux For Current ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc No_Coverage_Flux For Current ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Bc No_Coverage_Flux For Current ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc No_Coverage_Flux For Current ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Resistive_Load

Syntax

Bc No_Coverage_Flux For Current ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Resistive_Load ([Toggle = *toggle*] | [Area = *Area*] | [Referencevoltage = *ReferenceVoltage*])

Summary

Model a boundary attached to a resistive load. The local applied current density is V / (R A) where V is the local voltage, R is the specified resistance, and A is the area of the surface the BC is applied to.

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Area	real	-1
ReferenceVoltage	real	0

Scalar_String_Function

Syntax

Bc No_Coverage_Flux For Current ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f	"string"	–

User_Field

Syntax

Bc No_Coverage_Flux For Current ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc No_Coverage_Flux For Current ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc No_Coverage_Flux For Current ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

Cvfem_Continuity

Constant

Syntax

Bc No_Coverage_Flux For Cvfem_Continuity ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | Value = *value*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
value	real	–

Exponential

Syntax

Bc No_Coverage_Flux For Cvfem_Continuity ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc No_Coverage_Flux For Cvfem_Continuity ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
global_name	"string"	–
component	integer	0

Inflow

Syntax

Bc No_Coverage_Flux For Cvfem_Continuity ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Inflow [Toggle = *toggle*]

Summary

IBP continuity inflow; nodal interp

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–

Interface

Syntax

Bc No_Coverage_Flux For Cvfem_Continuity ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Interface [Toggle = *toggle*]

Summary

INTERFACE CVFEM continuity flux

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–

Mass_Flux

Syntax

Bc No_Coverage_Flux For Cvfem_Continuity ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Mass_Flux ([Toggle = *toggle*] | Total_Mdot = *total_mdot*)

Summary

Mass flux CVFEM continuity flux

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
total_mdot	real	–

Nd_Table

Syntax

Bc No_Coverage_Flux For Cvfem_Continuity ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Open_Flow

Syntax

Bc No_Coverage_Flux For Cvfem_Continuity ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Open_Flow ([Toggle = *toggle*] | [Pressure = *pressure*] | [Total_Pressure = *total_pressure*])

Summary

Open bc for continuity

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
pressure	real	0
total_pressure	real	0

Polynomial

Syntax

Bc No_Coverage_Flux For Cvfem_Continuity ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Porous_Robin_Averaged_Coeff

Syntax

Bc No_Coverage_Flux For Cvfem_Continuity ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Porous_Robin_Averaged_Coeff ([Toggle = *toggle*] | P_Field = *P_Field* | Massflux_Field = *MassFlux_Field* | Oppositecoeff_Field = *OppositeCoeff_Field*)

Summary

Boundary condition for loose coupling of continuity to a free fluid region

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
P_Field	"string"	–
MassFlux_Field	"string"	–
OppositeCoeff_Field	"string"	–

Porous_Robin_One_Region

Syntax

Bc No_Coverage_Flux For Cvfem_Continuity ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Porous_Robin_One_Region [Toggle = *toggle*]

Summary

Boundary condition for loose coupling of continuity to a free fluid region

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–

Scalar_String_Function

Syntax

Bc No_Coverage_Flux For Cvfem_Continuity ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
f	“string”	–

Simple_Inflow

Syntax

Bc No_Coverage_Flux For Cvfem_Continuity ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Simple_Inflow ([Toggle = *toggle*] | *{density | value}* = *Density* | [Velocity_X = *Velocity_X*] | [Velocity_Y = *Velocity_Y*] | [Velocity_Z = *Velocity_Z*])

Summary

IBP continuity inflow; specified dens and u

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Density	real	–
Velocity_X	real	0
Velocity_Y	real	0
Velocity_Z	real	0

User_Field

Syntax

Bc No_Coverage_Flux For Cvfem_Continuity ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc No_Coverage_Flux For Cvfem_Continuity ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc No_Coverage_Flux For Cvfem_Continuity ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

Cvfem_Density_Edge_Gradient_Projection

Constant

Syntax

Bc No_Coverage_Flux For Cvfem_Density_Edge_Gradient_Projection ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | Value = *value*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
value	real	–

Exponential

Syntax

Bc No_Coverage_Flux For Cvfem_Density_Edge_Gradient_Projection ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc No_Coverage_Flux For Cvfem_Density_Edge_Gradient_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Bc No_Coverage_Flux For Cvfem_Density_Edge_Gradient_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc No_Coverage_Flux For Cvfem_Density_Edge_Gradient_Projection ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc No_Coverage_Flux For Cvfem_Density_Edge_Gradient_Projection ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f	"string"	–

Simple_Interp

Syntax

Bc No_Coverage_Flux For Cvfem_Density_Edge_Gradient_Projection ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Simple_Interp [Toggle = *toggle*]

Summary

CVFEM edge-based projection simple interpolation flux BC

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–

User_Field

Syntax

Bc No_Coverage_Flux For Cvfem_Density_Edge_Gradient_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
scaling	real	1
global_var	“string”	–
time_function	“string”	–

User_Function

Syntax

Bc No_Coverage_Flux For Cvfem_Density_Edge_Gradient_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc No_Coverage_Flux For Cvfem_Density_Edge_Gradient_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

Cvfem_Dispersed_Continuity

Constant

Syntax

Bc No_Coverage_Flux For Cvfem_Dispersed_Continuity ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | Value = *value*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
value	real	–

Exponential

Syntax

Bc No_Coverage_Flux For Cvfem_Dispersed_Continuity ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc No_Coverage_Flux For Cvfem_Dispersed_Continuity ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
global_name	"string"	–
component	integer	0

Inflow

Syntax

Bc No_Coverage_Flux For Cvfem_Dispersed_Continuity ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Inflow [Toggle = *toggle*]

Summary

Inflow mass flux for dispersed phase continuity equation; nodal interpolation

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–

Nd_Table

Syntax

Bc No_Coverage_Flux For Cvfem_Dispersed_Continuity ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Open_Flow

Syntax

Bc No_Coverage_Flux For Cvfem_Dispersed_Continuity ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Open_Flow [Toggle = *toggle*]

Summary

Open bc for dispersed continuity

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–

Polynomial

Syntax

Bc No_Coverage_Flux For Cvfem_Dispersed_Continuity ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset,

X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc No_Coverage_Flux For Cvfem_Dispersed_Continuity ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be

a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f	"string"	–

Simple_Inflow

Syntax

Bc No_Coverage_Flux For Cvfem_Dispersed_Continuity ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Simple_Inflow ([Toggle = *toggle*] | [Velocity_X = *Velocity_X*] | [Velocity_Y = *Velocity_Y*] | [Velocity_Z = *Velocity_Z*])

Summary

Inflow mass flux for dispersed continuity equation; specified density and velocity

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Velocity_X	real	0
Velocity_Y	real	0
Velocity_Z	real	0

User_Field

Syntax

Bc No_Coverage_Flux For Cvfem_Dispersed_Continuity ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc No_Coverage_Flux For Cvfem_Dispersed_Continuity ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc No_Coverage_Flux For Cvfem_Dispersed_Continuity ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

Cvfem_Dispersed_Momentum

Constant

Syntax

Bc No_Coverage_Flux For Cvfem_Dispersed_Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | [Valuex = *valuex*] | [Valuey = *valuey*] | [Valuez = *valuez*])

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
valuex	real	0
valuey	real	0
valuez	real	0

Open_Adv_Flow

Syntax

Bc No_Coverage_Flux For Cvfem_Dispersed_Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Open_Adv_Flow ([Toggle = *toggle*] | [Coflow_Entrainment_Valuex = *Coflow_Entrainment_ValueX*] | [Coflow_Entrainment_Valuey = *Coflow_Entrainment_ValueY*] | [Coflow_Entrainment_Valuez = *Coflow_Entrainment_ValueZ*])

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Coflow_Entrainment_ValueX	real	0
Coflow_Entrainment_ValueY	real	0
Coflow_Entrainment_ValueZ	real	0

Open_Flow

Syntax

Bc No_Coverage_Flux For Cvfem_Dispersed_Momentum ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Open_Flow [Toggle = *toggle*]

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–

Symmetry_Flow

Syntax

Bc No_Coverage_Flux For Cvfem_Dispersed_Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Symmetry_Flow [Toggle = *toggle*]

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–

Vector_String_Function

Syntax

Bc No_Coverage_Flux For Cvfem_Dispersed_Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Vector_String_Function ([Toggle = *toggle*] | F_X = *f_x* | F_Y = *f_y* | [F_Z = *f_z*])

Summary

Value from a string function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is

active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f_x	"string"	–
f_y	"string"	–
f_z	"string"	–

Cvfem_Dispersed_Momentum_X

Constant

Syntax

Bc No_Coverage_Flux For Cvfem_Dispersed_Momentum_X ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | Value = *value*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
value	real	–

Exponential

Syntax

Bc No_Coverage_Flux For Cvfem_Dispersed_Momentum_X ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc No_Coverage_Flux For Cvfem_Dispersed_Momentum_X ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Bc No_Coverage_Flux For Cvfem_Dispersed_Momentum_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc No_Coverage_Flux For Cvfem_Dispersed_Momentum_X ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc No_Coverage_Flux For Cvfem_Dispersed_Momentum_X ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f	"string"	–

User_Field

Syntax

Bc No_Coverage_Flux For Cvfem_Dispersed_Momentum_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc No_Coverage_Flux For Cvfem_Dispersed_Momentum_X ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc No_Coverage_Flux For Cvfem_Dispersed_Momentum_X ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

Cvfem_Dispersed_Momentum_Y

Constant

Syntax

Bc No_Coverage_Flux For Cvfem_Dispersed_Momentum_Y ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | Value = *value*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
value	real	–

Exponential

Syntax

Bc No_Coverage_Flux For Cvfem_Dispersed_Momentum_Y ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc No_Coverage_Flux For Cvfem_Dispersed_Momentum_Y ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Bc No_Coverage_Flux For Cvfem_Dispersed_Momentum_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc No_Coverage_Flux For Cvfem_Dispersed_Momentum_Y ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc No_Coverage_Flux For Cvfem_Dispersed_Momentum_Y ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f	"string"	–

User_Field

Syntax

Bc No_Coverage_Flux For Cvfem_Dispersed_Momentum_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc No_Coverage_Flux For Cvfem_Dispersed_Momentum_Y ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc No_Coverage_Flux For Cvfem_Dispersed_Momentum_Y ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

Cvfem_Dispersed_Momentum_Z

Constant

Syntax

Bc No_Coverage_Flux For Cvfem_Dispersed_Momentum_Z ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | Value = *value*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
value	real	–

Exponential

Syntax

Bc No_Coverage_Flux For Cvfem_Dispersed_Momentum_Z ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc No_Coverage_Flux For Cvfem_Dispersed_Momentum_Z ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
global_name	“string”	–
component	integer	0

Nd_Table

Syntax

Bc No_Coverage_Flux For Cvfem_Dispersed_Momentum_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc No_Coverage_Flux For Cvfem_Dispersed_Momentum_Z ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc No_Coverage_Flux For Cvfem_Dispersed_Momentum_Z ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f	"string"	–

User_Field

Syntax

Bc No_Coverage_Flux For Cvfem_Dispersed_Momentum_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc No_Coverage_Flux For Cvfem_Dispersed_Momentum_Z ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc No_Coverage_Flux For Cvfem_Dispersed_Momentum_Z ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

Cvfem_Div_Projection

Constant

Syntax

Bc No_Coverage_Flux For Cvfem_Div_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | [Valuexx = *valuexx*] | [Valuexy = *valuexy*] | [Valuexz = *valuexz*] | [Valueyx = *valueyx*] | [Valueyy = *valueyy*] | [Valueyz = *valueyz*] | [Valuezx = *valuezx*] | [Valuezy = *valuezy*] | [Valuezz = *valuezz*] | [Valuet11 = *valuet11*] | [Valuet12 = *valuet12*] | [Valuet13 = *valuet13*] | [Valuet21 = *valuet21*] | [Valuet22 = *valuet22*] | [Valuet23 = *valuet23*] | [Valuet31 = *valuet31*] | [Valuet32 = *valuet32*] | [Valuet33 = *valuet33*])

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
valuexx	real	0
valuexy	real	0
valuexz	real	0
valueyx	real	0
valueyy	real	0
valueyz	real	0
valuezx	real	0
valuezy	real	0
valuezz	real	0
valuet11	real	0
valuet12	real	0
valuet13	real	0
valuet21	real	0
valuet22	real	0
valuet23	real	0
valuet31	real	0
valuet32	real	0
valuet33	real	0

Simple_Interp_Tensor

Syntax

Bc No_Coverage_Flux For Cvfem_Div_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Simple_Interp_Tensor ([Toggle = *toggle*] | Source = *source*)

Summary

CVFEM divergence projection simple interpolation tensor flux BC

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
source	"string"	–

Simple_Interp_Vector

Syntax

Bc No_Coverage_Flux For Cvfem_Div_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Simple_Interp_Vector ([Toggle = *toggle*] | Source = *source*)

Summary

CVFEM divergence projection simple interpolation vector flux BC

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
source	"string"	–

Tensor_String_Function

Syntax

Bc No_Coverage_Flux For Cvfem_Div_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Tensor_String_Function ([Toggle = *toggle*] | F_Xx = *f_xx* | F_Yy = *f_yy* | [F_Xy = *f_xy*] | [F_Zz = *f_zz*] | [F_Xz = *f_xz*] | [F_Yz = *f_yz*] | [F_Yx = *f_yx*] | [F_Zx = *f_zx*] | [F_Zy = *f_zy*])

Summary

Value from a string function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
f_xx	“string”	–
f_yy	“string”	–
f_xy	“string”	0.0
f_zz	“string”	–
f_xz	“string”	0.0
f_yz	“string”	0.0
f_yx	“string”	0.0
f_zx	“string”	0.0
f_zy	“string”	0.0

Tensor_String_Function_Symmetric

Syntax

Bc No_Coverage_Flux For Cvfem_Div_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Tensor_String_Function_Symmetric ([Toggle = *toggle*] | F_Xx = *f_xx* | F_Yy = *f_yy* | [F_Xy = *f_xy*] | [F_Zz = *f_zz*] | [F_Xz = *f_xz*] | [F_Yz = *f_yz*])

Summary

Value from a string function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f_xx	"string"	–
f_yy	"string"	–
f_xy	"string"	0.0
f_zz	"string"	–
f_xz	"string"	0.0
f_yz	"string"	0.0

Cvfem_Energy

Constant

Syntax

Bc No_Coverage_Flux For Cvfem_Energy ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Data = *data*] | *{flux | value}* = *flux*)

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
data	"string"	–
flux	real	–

Exponential

Syntax

Bc No_Coverage_Flux For Cvfem_Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Data = *data*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
data	“string”	–
variable	“string”	–
constant	real	0
multiplier	real	1
exponent	real	–

Exponential_Vapor_Cooling

Syntax

Bc No_Coverage_Flux For Cvfem_Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential_Vapor_Cooling ([Data = *data*] | Tboil = *tboil* | L = *l* | Pa = *pa* | Mw = *mw* | Alpha = *alpha* | [Tl = *tl*] | [Th = *th*])

Summary

Exponential vapor cooling energy flux

Description

Flux is given by:

$$F = P_a L\alpha \exp\left(\frac{L}{R}\left(\frac{1}{T_{boil}} - \frac{1}{T}\right)\right)\sqrt{\frac{1}{2\pi RT}}$$

where R is the specific gas constant ($\bar{R}$/MW). For T less than T_L, zero flux is applied. For T between T_L and T_H, a 3rd order smoothing polynomial is used to provide continuity of values and derivatives between 0 and the expression given above. For T greater than T_H, the above expression is used. If not provided, T_L and T_H are set to 10% below and above T_{Boil}, respectively.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
data	"string"	–
tboil	real	–
l	real	–
pa	real	–
mw	real	–
alpha	real	–
tl	real	-1
th	real	-1

Generalized_Nat_Conv

Syntax

Bc No_Coverage_Flux For Cvfem_Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Generalized_Nat_Conv ([Data = *data*] | [Multiplier = *multiplier*] | [H = *h*] | [T_Ref = *t_ref*])

Summary

CVFEM energy generalized natural convection flux BC

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
data	"string"	–
multiplier	real	1
h	"string"	–
t_ref	"string"	–

Generalized_Rad

Syntax

Bc No_Coverage_Flux For Cvfem_Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Generalized_Rad ([Data = *data*] | [Multiplier = *multiplier*] | [T_Ref = *t_ref*])

Summary

Radiative energy flux, with values for emissivity, form factor and irradiation taken from expressions

Description

Flux is:

$$M\epsilon f\left(\sigma T^4 - I\right)$$

M is MULTIPLIER. σ is the Stefan-Boltzmann constant. ϵ and F are expressions for emissivity and form factor. T is surface temperature. I is the expression for irradiation. If I does not already exist, it will be constructed as σT_{ref}^4, where T_{ref} is either provided by the BC Rad Reference Temperature expression or the inline T_{ref} parameter (constant or string function).

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
data	"string"	–
multiplier	real	1
t_ref	"string"	–

Global

Syntax

Bc No_Coverage_Flux For Cvfem_Energy ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Data = *data*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
data	"string"	–
global_name	"string"	–
component	integer	0

Interface

Syntax

Bc No_Coverage_Flux For Cvfem_Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Interface [Data = *data*]

Summary

INTERFACE CVFEM energy flux

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
data	“string”	–

Mass_Flux

Syntax

Bc No_Coverage_Flux For Cvfem_Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Mass_Flux ([Data = *data*] | Inflow_Phi = *Inflow_Phi* | Total_Mdot = *total_mdot*)

Summary

CVFEM energy mass flux BC

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
data	"string"	–
Inflow_Phi	real	–
total_mdot	real	–

Mass_Flux_From_Temperature

Syntax

Bc No_Coverage_Flux For Cvfem_Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Mass_Flux_From_Temperature ([Data = *data*] | [Temperature = *temperature*] | [Pressure = *pressure*] | [Mass_Fraction_0 = *mass_fraction_0*] | [Mass_Fraction_1 = *mass_fraction_1*] | [Mass_Fraction_2 = *mass_fraction_2*] | [Mass_Fraction_3 = *mass_fraction_3*] | [Mass_Fraction_4 = *mass_fraction_4*] | [Mass_Fraction_5 = *mass_fraction_5*] | [Mass_Fraction_6 = *mass_fraction_6*] | [Mass_Fraction_7 = *mass_fraction_7*] | [Mass_Fraction_8 = *mass_fraction_8*] | Total_Mdot = *total_mdot*)

Summary

CVFEM energy mass flux from temperature flux BC

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
data	“string”	–
temperature	real	0
pressure	real	0
mass_fraction_0	real	0
mass_fraction_1	real	0
mass_fraction_2	real	0
mass_fraction_3	real	0
mass_fraction_4	real	0
mass_fraction_5	real	0
mass_fraction_6	real	0
mass_fraction_7	real	0
mass_fraction_8	real	0
total_mdot	real	–

Nat_Conv

Syntax

Bc No_Coverage_Flux For Cvfem_Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nat_Conv ([Data = *data*] | H = *h* | T_Ref = *t_ref*)

Summary

CVFEM convective energy flux defined with constant models for heat transfer coefficient and reference temperature

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
data	"string"	–
h	real	–
t_ref	real	–

Nd_Table

Syntax

Bc No_Coverage_Flux For Cvfem_Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Data = *data*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
data	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Open_Adv_Flow

Syntax

Bc No_Coverage_Flux For Cvfem_Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Open_Adv_Flow ([Data = *data*] | Far_Field_Entrainment_Value = *Far_Field_Entrainment_Value* | [Pressure = *pressure*] | [Total_Pressure = *total_pressure*])

Summary

CVFEM energy open advective flow flux

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
data	“string”	–
Far_Field_Entrainment_Value	real	–
pressure	real	0
total_pressure	real	0

Open_Adv_Flow_From_Temperature

Syntax

Bc No_Coverage_Flux For Cvfem_Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Open_Adv_Flow_From_Temperature ([Data = *data*] | [Temperature = *temperature*] | [Pressure = *pressure*] | [Mass_Fraction_0 = *mass_fraction_0*] | [Mass_Fraction_1 = *mass_fraction_1*] | [Mass_Fraction_2 = *mass_fraction_2*] | [Mass_Fraction_3 = *mass_fraction_3*] | [Mass_Fraction_4 = *mass_fraction_4*] | [Mass_Fraction_5 = *mass_fraction_5*] | [Mass_Fraction_6 = *mass_fraction_6*] | [Mass_Fraction_7 = *mass_fraction_7*] | [Mass_Fraction_8 = *mass_fraction_8*] | [Total_Pressure = *total_pressure*])

Summary

CVFEM energy open advective flow from temperature flux

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
data	“string”	–
temperature	real	0
pressure	real	0
mass_fraction_0	real	0
mass_fraction_1	real	0
mass_fraction_2	real	0
mass_fraction_3	real	0
mass_fraction_4	real	0
mass_fraction_5	real	0
mass_fraction_6	real	0
mass_fraction_7	real	0
mass_fraction_8	real	0
total_pressure	real	0

Polynomial

Syntax

Bc No_Coverage_Flux For Cvfem_Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Data = *data*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
data	“string”	–
variable	“string”	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Porous_Robin_Coupled

Syntax

Bc No_Coverage_Flux For Cvfem_Energy ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Porous_Robin_Coupled ([Data = *data*] | Enthalpy_Field = *enthalpy_field* | Enthalpy_Diffusive_Flux_Field = *enthalpy_diffusive_flux_field* | Averaged_Coeff_Field = *averaged_coeff_field*)

Summary

Boundary condition for loose coupling of enthalpy to a porous region

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
data	“string”	–
enthalpy_field	“string”	–
enthalpy_diffusive_flux_field	“string”	–
averaged_coeff_field	“string”	–

Porous_Robin_Coupled_One_Region

Syntax

Bc No_Coverage_Flux For Cvfem_Energy ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Porous_Robin_Coupled_One_Region [Data = *data*]

Summary

Boundary condition for loose coupling of enthalpy to a porous region

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
data	“string”	–

Porous_Robin_Coupled_With_Solid_Phase_Convection

Syntax

Bc No_Coverage_Flux For Cvfem_Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Porous_Robin_Coupled_With_Solid_Phase_Convection ([Data = *data*] | [Multiplier = *multiplier*] | Enthalpy_Field = *enthalpy_field* | Enthalpy_Diffusive_Flux_Field = *enthalpy_diffusive_flux_field* | Solid_Phase_Temperature_Field = *solid_phase_temperature_field* | Porosity_Field = *porosity_field* | Averaged_Coeff_Field = *averaged_coeff_field* | Volumetric_Heat_Transfer_Coeff_Field = *volumetric_heat_transfer_coeff_field* | Specific_Surface_Area_Field = *specific_surface_area_field*)

Summary

Boundary condition for loose coupling of enthalpy to a porous region

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
data	“string”	–
multiplier	real	1
enthalpy_field	“string”	–
enthalpy_diffusive_flux_field	“string”	–
solid_phase_temperature_field	“string”	–
porosity_field	“string”	–
averaged_coeff_field	“string”	–
volumetric_heat_transfer_coeff_field	“string”	–
specific_surface_area_field	“string”	–

Porous_Robin_Coupled_With_Solid_Phase_Convection_One_Region

Syntax

Bc No_Coverage_Flux For Cvfem_Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Porous_Robin_Coupled_With_Solid_Phase_Convection_One_Region ([Data = *data*] | [Multiplier = *multiplier*])

Summary

Boundary condition for coupling of enthalpy to a porous region in the same region.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
data	“string”	–
multiplier	real	1

Rad

Syntax

Bc No_Coverage_Flux For Cvfem_Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Rad ([Data = *data*] | Crad = *crad* | T_Ref = *t_ref*)

Summary

Radiative energy flux defined with constant models for emissivity and reference temperature.

Description

The applied flux is given by:

$$c_{rad}\left(T^4 - T_{ref}^4\right)$$

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
data	"string"	–
crad	real	–
t_ref	real	–

Ray_Trace

Syntax

Bc No_Coverage_Flux For Cvfem_Energy ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Ray_Trace ([Data = *data*] | Ray_Source = *ray_source* | [Emission = *emission*])

Summary

Apply a ray tracing energy flux

Description

Adds an energy flux based on the intersection of rays defined in a ray tracing source block for the equation system with name given by the RAY_SOURCE parameter with the surface. The optional EMISSION parameter also imposes a σT^4 blackbody radiative emission term

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
data	"string"	–
ray_source	"string"	–
emission	"string"	FALSE

Scalar_String_Function

Syntax

Bc No_Coverage_Flux For Cvfem_Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Data = *data*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
data	“string”	–
f	“string”	–

User_Field

Syntax

Bc No_Coverage_Flux For Cvfem_Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Data = *data*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
data	“string”	–
name	“string”	–
scaling	real	1
global_var	“string”	–
time_function	“string”	–

User_Function

Syntax

Bc No_Coverage_Flux For Cvfem_Energy ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Data = *data*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
data	“string”	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Bc No_Coverage_Flux For Cvfem_Energy ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Data = *data*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
data	“string”	–
Name	“string”	–
plugin_parameters	“string”	–

Wall_Function_From_Cht_Temperature

Syntax

Bc No_Coverage_Flux For Cvfem_Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Wall_Function_From_Cht_Temperature ([Data = *data*] | Wall_Friction_Factor = *wall_friction_factor*)

Summary

CVFEM energy wall function from conjugate heat transfer temperature flux BC

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
data	“string”	–
wall_friction_factor	real	–

Wall_Function_From_Temperature

Syntax

Bc No_Coverage_Flux For Cvfem_Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Wall_Function_From_Temperature ([Data = *data*] | *{t | value}* = *t* | Wall_Friction_Factor = *wall_friction_factor*)

Summary

CVFEM energy wall function from temperature flux BC

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
data	"string"	–
t	real	–
wall_friction_factor	real	–

Cvfem_Level_Set

Open_Adv_Flow

Syntax

Bc No_Coverage_Flux For Cvfem_Level_Set ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Open_Adv_Flow ([Toggle = *toggle*] | [Pressure = *pressure*] | [Total_Pressure = *total_pressure*] | Far_Field_Entrainment_Value = *far_field_entrainment_value*)

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
pressure	real	0
total_pressure	real	0
far_field_entrainment_value	real	–

Open_Nc_Adv_Flow

Syntax

Bc No_Coverage_Flux For Cvfem_Level_Set ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Open_Nc_Adv_Flow ([Toggle = *toggle*] | Far_Field_Entrainment_Value = *far_field_entrainment_value*)

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
far_field_entrainment_value	real	–

Cvfem_Lumped_Div_Projection

Constant

Syntax

Bc No_Coverage_Flux For Cvfem_Lumped_Div_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | [Valuexx = *valuexx*] | [Valuexy = *valuexy*] | [Valuexz = *valuexz*] | [Valueyx = *valueyx*] | [Valueyy = *valueyy*] | [Valueyz = *valueyz*] | [Valuezx = *valuezx*] | [Valuezy = *valuezy*] | [Valuezz = *valuezz*] | [Valuet11 = *valuet11*] | [Valuet12 = *valuet12*] | [Valuet13 = *valuet13*] | [Valuet21 = *valuet21*] | [Valuet22 = *valuet22*] | [Valuet23 = *valuet23*] | [Valuet31 = *valuet31*] | [Valuet32 = *valuet32*] | [Valuet33 = *valuet33*])

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
valuexx	real	0
valuexy	real	0
valuexz	real	0
valueyx	real	0
valueyy	real	0
valueyz	real	0
valuezx	real	0
valuezy	real	0
valuezz	real	0
valuet11	real	0
valuet12	real	0
valuet13	real	0
valuet21	real	0
valuet22	real	0
valuet23	real	0
valuet31	real	0
valuet32	real	0
valuet33	real	0

Simple_Interp_Tensor

Syntax

Bc No_Coverage_Flux For Cvfem_Lumped_Div_Projection ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Simple_Interp_Tensor ([Toggle = *toggle*] | Source = *source*)

Summary

CVFEM lumped divergence projection simple interpolation tensor flux BC

Description

The following options are present in most boundary conditions/sources. The

meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
source	"string"	–

Simple_Interp_Vector

Syntax

Bc No_Coverage_Flux For Cvfem_Lumped_Div_Projection ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Simple_Interp_Vector ([Toggle = *toggle*] | Source = *source*)

Summary

CVFEM lumped divergence projection simple interpolation vector flux BC

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
source	"string"	–

Tensor_String_Function

Syntax

Bc No_Coverage_Flux For Cvfem_Lumped_Div_Projection ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Tensor_String_Function ([Toggle = *toggle*] | F_Xx = *f_xx* | F_Yy = *f_yy* | [F_Xy = *f_xy*] | [F_Zz = *f_zz*] | [F_Xz = *f_xz*] | [F_Yz = *f_yz*] | [F_Yx = *f_yx*] | [F_Zx = *f_zx*] | [F_Zy = *f_zy*])

Summary

Value from a string function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
f_xx	“string”	–
f_yy	“string”	–
f_xy	“string”	0.0
f_zz	“string”	–
f_xz	“string”	0.0
f_yz	“string”	0.0
f_yx	“string”	0.0
f_zx	“string”	0.0
f_zy	“string”	0.0

Tensor_String_Function_Symmetric

Syntax

Bc No_Coverage_Flux For Cvfem_Lumped_Div_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Tensor_String_Function_Symmetric ([Toggle = *toggle*] | F_Xx = *f_xx* | F_Yy = *f_yy* | [F_Xy = *f_xy*] | [F_Zz = *f_zz*] | [F_Xz = *f_xz*] | [F_Yz = *f_yz*])

Summary

Value from a string function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f_xx	"string"	–
f_yy	"string"	–
f_xy	"string"	0.0
f_zz	"string"	–
f_xz	"string"	0.0
f_yz	"string"	0.0

Cvfem_Lumped_Projection

Constant

Syntax

Bc No_Coverage_Flux For Cvfem_Lumped_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | Value = *value*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
value	real	–

Exponential

Syntax

Bc No_Coverage_Flux For Cvfem_Lumped_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc No_Coverage_Flux For Cvfem_Lumped_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Bc No_Coverage_Flux For Cvfem_Lumped_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Open

Syntax

Bc No_Coverage_Flux For Cvfem_Lumped_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Open ([Toggle = *toggle*] | [Total_Pressure = *total_pressure*] | [Pressure = *pressure*])

Summary

CVFEM lumped projection open flux BC

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
total_pressure	real	0
pressure	real	0

Polynomial

Syntax

Bc No_Coverage_Flux For Cvfem_Lumped_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc No_Coverage_Flux For Cvfem_Lumped_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f	"string"	–

Simple_Interp

Syntax

Bc No_Coverage_Flux For Cvfem_Lumped_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Simple_Interp [Toggle = *toggle*]

Summary

CVFEM lumped projection simple interpolation flux BC

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–

User_Field

Syntax

Bc No_Coverage_Flux For Cvfem_Lumped_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
scaling	real	1
global_var	“string”	–
time_function	“string”	–

User_Function

Syntax

Bc No_Coverage_Flux For Cvfem_Lumped_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc No_Coverage_Flux For Cvfem_Lumped_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

Cvfem_Mass_Fraction

Constant

Syntax

Bc No_Coverage_Flux For Cvfem_Mass_Fraction ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | Value = *value*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
value	real	–

Exponential

Syntax

Bc No_Coverage_Flux For Cvfem_Mass_Fraction ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc No_Coverage_Flux For Cvfem_Mass_Fraction ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
global_name	"string"	–
component	integer	0

Mass_Flux

Syntax

Bc No_Coverage_Flux For Cvfem_Mass_Fraction ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Mass_Flux ([Toggle = *toggle*] | Total_Mdot = *total_mdot* | [Inflow_Phi = *inflow_phi*])

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
total_mdot	real	–
inflow_phi	real	0

Nd_Table

Syntax

Bc No_Coverage_Flux For Cvfem_Mass_Fraction ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Open_Adv_Flow

Syntax

Bc No_Coverage_Flux For Cvfem_Mass_Fraction ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Open_Adv_Flow ([Toggle = *toggle*] | [Pressure = *pressure*] | [Total_Pressure = *total_pressure*] | [Far_Field_Entrainment_Value = *far_field_entrainment_value*])

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
pressure	real	0
total_pressure	real	0
far_field_entrainment_value	real	0

Polynomial

Syntax

Bc No_Coverage_Flux For Cvfem_Mass_Fraction ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
variable	“string”	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Porous_Robin_Coupled

Syntax

Bc No_Coverage_Flux For Cvfem_Mass_Fraction ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Porous_Robin_Coupled ([Toggle = *toggle*] | Mass_Fraction_Field = *mass_fraction_field* | Mass_Fraction_Diffusive_Flux_Field = *mass_fraction_diffusive_flux_field* | Averaged_Coeff_Field = *averaged_coeff_field*)

Summary

Boundary condition for loose coupling of scalars to a free fluid region

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
mass_fraction_field	“string”	–
mass_fraction_diffusive_flux_field	“string”	–
averaged_coeff_field	“string”	–

Porous_Robin_Coupled_One_Region

Syntax

Bc No_Coverage_Flux For Cvfem_Mass_Fraction ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Porous_Robin_Coupled_One_Region [Toggle = *toggle*]

Summary

Boundary condition for loose coupling of scalars to a free fluid region

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–

Scalar_String_Function

Syntax

Bc No_Coverage_Flux For Cvfem_Mass_Fraction ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
f	“string”	–

User_Field

Syntax

Bc No_Coverage_Flux For Cvfem_Mass_Fraction ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc No_Coverage_Flux For Cvfem_Mass_Fraction ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc No_Coverage_Flux For Cvfem_Mass_Fraction ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
Name	“string”	–
plugin_parameters	“string”	–

Cvfem_Mixture_Fraction

Constant

Syntax

Bc No_Coverage_Flux For Cvfem_Mixture_Fraction ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | Value = *value*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
value	real	–

Exponential

Syntax

Bc No_Coverage_Flux For Cvfem_Mixture_Fraction ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Free_Open_Flow

Syntax

Bc No_Coverage_Flux For Cvfem_Mixture_Fraction ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Free_Open_Flow [Toggle = *toggle*]

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–

Global

Syntax

Bc No_Coverage_Flux For Cvfem_Mixture_Fraction ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
global_name	“string”	–
component	integer	0

Mass_Flux

Syntax

Bc No_Coverage_Flux For Cvfem_Mixture_Fraction ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Mass_Flux ([Toggle = *toggle*] | Total_Mdot = *total_mdot* | [Inflow_Phi = *inflow_phi*])

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
total_mdot	real	–
inflow_phi	real	0

Nd_Table

Syntax

Bc No_Coverage_Flux For Cvfem_Mixture_Fraction ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Open_Adv_Flow

Syntax

Bc No_Coverage_Flux For Cvfem_Mixture_Fraction ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Open_Adv_Flow ([Toggle = *toggle*] | [Pressure = *pressure*] | [Total_Pressure = *total_pressure*] | [Far_Field_Entrainment_Value = *far_field_entrainment_value*])

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
pressure	real	0
total_pressure	real	0
far_field_entrainment_value	real	0

Polynomial

Syntax

Bc No_Coverage_Flux For Cvfem_Mixture_Fraction ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc No_Coverage_Flux For Cvfem_Mixture_Fraction ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f	"string"	–

User_Field

Syntax

Bc No_Coverage_Flux For Cvfem_Mixture_Fraction ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc No_Coverage_Flux For Cvfem_Mixture_Fraction ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc No_Coverage_Flux For Cvfem_Mixture_Fraction ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

Cvfem_Momentum

Constant

Syntax

Bc No_Coverage_Flux For Cvfem_Momentum ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | [Valuex = *valuex*] | [Valuey = *valuey*] | [Valuez = *valuez*])

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
valuex	real	0
valuey	real	0
valuez	real	0

Constant_Traction

Syntax

Bc No_Coverage_Flux For Cvfem_Momentum ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant_Traction ([Toggle = *toggle*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
x	real	0
y	real	0
z	real	0

Exponential_Vapor_Recoil_Pressure

Syntax

Bc No_Coverage_Flux For Cvfem_Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential_Vapor_Recoil_Pressure ([Toggle = *toggle*] | Tboil = *tboil* | L = *l* | Pa = *pa* | [Beta = *beta*] | Mw = *mw* | [Tl = *tl*] | [Th = *th*])

Summary

Exponential vapor recoil pressure momentum flux

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
tboil	real	–
l	real	–
pa	real	–
beta	real	1
mw	real	–
tl	real	-1
th	real	-1

Free_Open_Flow

Syntax

Bc No_Coverage_Flux For Cvfem_Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Free_Open_Flow ([Toggle = *toggle*] | Pressure = *pressure*)

Summary

Open flow for momentum; apply the full stress

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
pressure	real	–

Interface

Syntax

Bc No_Coverage_Flux For Cvfem_Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Interface [Toggle = *toggle*]

Summary

INTERFACE CVFEM momentum flux

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–

Mass_Flux

Syntax

Bc No_Coverage_Flux For Cvfem_Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Mass_Flux ([Toggle = *toggle*] | Total_Mdot = *total_mdot* | [Temperature = *temperature*] | [Pressure = *pressure*] | [Mass_Fraction_0 = *mass_fraction_0*] | [Mass_Fraction_1 = *mass_fraction_1*] | [Mass_Fraction_2 = *mass_fraction_2*] | [Mass_Fraction_3 = *mass_fraction_3*] | [Mass_Fraction_4 = *mass_fraction_4*] | [Mass_Fraction_5 = *mass_fraction_5*] | [Mass_Fraction_6 = *mass_fraction_6*] | [Mass_Fraction_7 = *mass_fraction_7*] | [Mass_Fraction_8 = *mass_fraction_8*])

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
total_mdot	real	–
temperature	real	0
pressure	real	0
mass_fraction_0	real	0
mass_fraction_1	real	0
mass_fraction_2	real	0
mass_fraction_3	real	0
mass_fraction_4	real	0
mass_fraction_5	real	0
mass_fraction_6	real	0
mass_fraction_7	real	0
mass_fraction_8	real	0

Open_Adv_Flow

Syntax

Bc No_Coverage_Flux For Cvfem_Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Open_Adv_Flow ([Toggle = *toggle*] | [Pressure = *pressure*] | [Total_Pressure = *total_pressure*] | [Coflow_Entrainment_Value_X = *coflow_entrainment_value_x*] | [Coflow_Entrainment_Value_Y = *coflow_entrainment_value_y*] | [Coflow_Entrainment_Value_Z = *coflow_entrainment_value_z*])

Summary

CVFEM momentum open advective flow flux BC

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is

active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
pressure	real	0
total_pressure	real	0
coflow_entrainment_value_x	real	0
coflow_entrainment_value_y	real	0
coflow_entrainment_value_z	real	0

Open_Flow

Syntax

Bc No_Coverage_Flux For Cvfem_Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Open_Flow ([Toggle = *toggle*] | [Pressure = *pressure*] | [Total_Pressure = *total_pressure*])

Summary

Open flow for momentum; enforce normal gradient to be zero

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
pressure	real	0
total_pressure	real	0

Slip

Syntax

Bc No_Coverage_Flux For Cvfem_Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Slip ([Toggle = *toggle*] | [Vs_X = *vs_x*] | [Vs_Y = *vs_y*] | [Vs_Z = *vs_z*] | Beta = *beta*)

Summary

Slip momentum flux

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
vs_x	real	0
vs_y	real	0
vs_z	real	0
beta	real	–

Slip_Length

Syntax

Bc No_Coverage_Flux For Cvfem_Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Slip_Length ([Toggle = *toggle*] | [Vs_X = *vs_x*] | [Vs_Y = *vs_y*] | [Vs_Z = *vs_z*] | Beta = *beta*)

Summary

Slip length momentum flux

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
vs_x	real	0
vs_y	real	0
vs_z	real	0
beta	real	–

Symmetry_Flow

Syntax

Bc No_Coverage_Flux For Cvfem_Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Symmetry_Flow [Toggle = *toggle*]

Summary

CVFEM momentum symmetry flow flux BC

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–

Vector_String_Function

Syntax

Bc No_Coverage_Flux For Cvfem_Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Vector_String_Function ([Toggle = *toggle*] | F_X = *f_x* | F_Y = *f_y* | [F_Z = *f_z*])

Summary

Value from a string function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f_x	"string"	–
f_y	"string"	–
f_z	"string"	–

Wall_Function

Syntax

Bc No_Coverage_Flux For Cvfem_Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Wall_Function ([Toggle = *toggle*] | [Wall_Velocity_X = *wall_velocity_x*] | [Wall_Velocity_Y = *wall_velocity_y*] | [Wall_Velocity_Z = *wall_velocity_z*] | Wall_Friction_Factor = *wall_friction_factor*)

Summary

CVFEM momentum wall function flux BC

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
wall_velocity_x	real	0
wall_velocity_y	real	0
wall_velocity_z	real	0
wall_friction_factor	real	–

Cvfem_Momentum_X

Constant

Syntax

Bc No_Coverage_Flux For Cvfem_Momentum_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent*

Name ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | Value = *value*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
value	real	–

Exponential

Syntax

Bc No_Coverage_Flux For Cvfem_Momentum_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc No_Coverage_Flux For Cvfem_Momentum_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Bc No_Coverage_Flux For Cvfem_Momentum_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc No_Coverage_Flux For Cvfem_Momentum_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc No_Coverage_Flux For Cvfem_Momentum_X ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f	"string"	–

User_Field

Syntax

Bc No_Coverage_Flux For Cvfem_Momentum_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc No_Coverage_Flux For Cvfem_Momentum_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc No_Coverage_Flux For Cvfem_Momentum_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

Cvfem_Momentum_Y

Constant

Syntax

Bc No_Coverage_Flux For Cvfem_Momentum_Y ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | Value = *value*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
value	real	–

Exponential

Syntax

Bc No_Coverage_Flux For Cvfem_Momentum_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc No_Coverage_Flux For Cvfem_Momentum_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Bc No_Coverage_Flux For Cvfem_Momentum_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc No_Coverage_Flux For Cvfem_Momentum_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc No_Coverage_Flux For Cvfem_Momentum_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f	"string"	–

User_Field

Syntax

Bc No_Coverage_Flux For Cvfem_Momentum_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc No_Coverage_Flux For Cvfem_Momentum_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc No_Coverage_Flux For Cvfem_Momentum_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

Cvfem_Momentum_Z

Constant

Syntax

Bc No_Coverage_Flux For Cvfem_Momentum_Z ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | Value = *value*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
value	real	–

Exponential

Syntax

Bc No_Coverage_Flux For Cvfem_Momentum_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc No_Coverage_Flux For Cvfem_Momentum_Z ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Bc No_Coverage_Flux For Cvfem_Momentum_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc No_Coverage_Flux For Cvfem_Momentum_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc No_Coverage_Flux For Cvfem_Momentum_Z ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f	"string"	–

User_Field

Syntax

Bc No_Coverage_Flux For Cvfem_Momentum_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc No_Coverage_Flux For Cvfem_Momentum_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc No_Coverage_Flux For Cvfem_Momentum_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

Cvfem_Pressure_Edge_Gradient_Projection

Constant

Syntax

Bc No_Coverage_Flux For Cvfem_Pressure_Edge_Gradient_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | Value = *value*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
value	real	–

Exponential

Syntax

Bc No_Coverage_Flux For Cvfem_Pressure_Edge_Gradient_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc No_Coverage_Flux For Cvfem_Pressure_Edge_Gradient_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
global_name	“string”	–
component	integer	0

Nd_Table

Syntax

Bc No_Coverage_Flux For Cvfem_Pressure_Edge_Gradient_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc No_Coverage_Flux For Cvfem_Pressure_Edge_Gradient_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc No_Coverage_Flux For Cvfem_Pressure_Edge_Gradient_Projection ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f	"string"	–

Simple_Interp

Syntax

Bc No_Coverage_Flux For Cvfem_Pressure_Edge_Gradient_Projection ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Simple_Interp [Toggle = *toggle*]

Summary

CVFEM edge-based projection simple interpolation flux BC

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–

User_Field

Syntax

Bc No_Coverage_Flux For Cvfem_Pressure_Edge_Gradient_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
scaling	real	1
global_var	“string”	–
time_function	“string”	–

User_Function

Syntax

Bc No_Coverage_Flux For Cvfem_Pressure_Edge_Gradient_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc No_Coverage_Flux For Cvfem_Pressure_Edge_Gradient_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

Cvfem_Projection

Constant

Syntax

Bc No_Coverage_Flux For Cvfem_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | Value = *value*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
value	real	–

Exponential

Syntax

Bc No_Coverage_Flux For Cvfem_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc No_Coverage_Flux For Cvfem_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Bc No_Coverage_Flux For Cvfem_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Open

Syntax

Bc No_Coverage_Flux For Cvfem_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Open ([Toggle = *toggle*] | [Total_Pressure = *total_pressure*] | [Pressure = *pressure*])

Summary

CVFEM projection open flux BC

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
total_pressure	real	0
pressure	real	0

Polynomial

Syntax

Bc No_Coverage_Flux For Cvfem_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc No_Coverage_Flux For Cvfem_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f	"string"	–

Simple_Interp

Syntax

Bc No_Coverage_Flux For Cvfem_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Simple_Interp [Toggle = *toggle*]

Summary

CVFEM projection simple interpolation flux BC

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–

User_Field

Syntax

Bc No_Coverage_Flux For Cvfem_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
scaling	real	1
global_var	“string”	–
time_function	“string”	–

User_Function

Syntax

Bc No_Coverage_Flux For Cvfem_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc No_Coverage_Flux For Cvfem_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

Cvfem_Sdr_Edge_Gradient_Projection

Constant

Syntax

Bc No_Coverage_Flux For Cvfem_Sdr_Edge_Gradient_Projection ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | Value = *value*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
value	real	–

Exponential

Syntax

Bc No_Coverage_Flux For Cvfem_Sdr_Edge_Gradient_Projection ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc No_Coverage_Flux For Cvfem_Sdr_Edge_Gradient_Projection ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Bc No_Coverage_Flux For Cvfem_Sdr_Edge_Gradient_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc No_Coverage_Flux For Cvfem_Sdr_Edge_Gradient_Projection ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc No_Coverage_Flux For Cvfem_Sdr_Edge_Gradient_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f	"string"	–

Simple_Interp

Syntax

Bc No_Coverage_Flux For Cvfem_Sdr_Edge_Gradient_Projection ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Simple_Interp [Toggle = *toggle*]

Summary

CVFEM edge-based projection simple interpolation flux BC

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–

User_Field

Syntax

Bc No_Coverage_Flux For Cvfem_Sdr_Edge_Gradient_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
scaling	real	1
global_var	“string”	–
time_function	“string”	–

User_Function

Syntax

Bc No_Coverage_Flux For Cvfem_Sdr_Edge_Gradient_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc No_Coverage_Flux For Cvfem_Sdr_Edge_Gradient_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

Cvfem_Solvent_Continuity

Constant

Syntax

Bc No_Coverage_Flux For Cvfem_Solvent_Continuity ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | Value = *value*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
value	real	–

Exponential

Syntax

Bc No_Coverage_Flux For Cvfem_Solvent_Continuity ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc No_Coverage_Flux For Cvfem_Solvent_Continuity ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
global_name	"string"	–
component	integer	0

Inflow

Syntax

Bc No_Coverage_Flux For Cvfem_Solvent_Continuity ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Inflow [Toggle = *toggle*]

Summary

Inflow mass flux for continuity equation; nodal interpolation

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–

Nd_Table

Syntax

Bc No_Coverage_Flux For Cvfem_Solvent_Continuity ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Open_Flow

Syntax

Bc No_Coverage_Flux For Cvfem_Solvent_Continuity ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Open_Flow ([Toggle = *toggle*] | [Pressure = *pressure*] | [Total_Pressure = *total_pressure*])

Summary

Open bc for continuity

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
pressure	real	0
total_pressure	real	0

Polynomial

Syntax

Bc No_Coverage_Flux For Cvfem_Solvent_Continuity ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
variable	“string”	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc No_Coverage_Flux For Cvfem_Solvent_Continuity ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f	"string"	–

Simple_Inflow

Syntax

Bc No_Coverage_Flux For Cvfem_Solvent_Continuity ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Simple_Inflow ([Toggle = *toggle*] | *{density | value}* = *Density* | [Velocity_X = *Velocity_X*] | [Velocity_Y = *Velocity_Y*] | [Velocity_Z = *Velocity_Z*])

Summary

Inflow mass flux for continuity equation; specified density and velocity

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Density	real	–
Velocity_X	real	0
Velocity_Y	real	0
Velocity_Z	real	0

User_Field

Syntax

Bc No_Coverage_Flux For Cvfem_Solvent_Continuity ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc No_Coverage_Flux For Cvfem_Solvent_Continuity ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc No_Coverage_Flux For Cvfem_Solvent_Continuity ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

Cvfem_Solvent_Momentum

Constant

Syntax

Bc No_Coverage_Flux For Cvfem_Solvent_Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | [Valuex = *valuex*] | [Valuey = *valuey*] | [Valuez = *valuez*])

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
valuex	real	0
valuey	real	0
valuez	real	0

Open_Adv_Flow

Syntax

Bc No_Coverage_Flux For Cvfem_Solvent_Momentum ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Open_Adv_Flow ([Toggle = *toggle*] | [Pressure = *pressure*] | [Total_Pressure = *total_pressure*] | [Coflow_Entrainment_Value_X = *coflow_entrainment_value_x*] | [Coflow_Entrainment_Value_Y = *coflow_entrainment_value_y*] | [Coflow_Entrainment_Value_Z = *coflow_entrainment_value_z*])

Summary

CVFEM solvent momentum open advective flow flux BC

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
pressure	real	0
total_pressure	real	0
coflow_entrainment_value_x	real	0
coflow_entrainment_value_y	real	0
coflow_entrainment_value_z	real	0

Open_Flow

Syntax

Bc No_Coverage_Flux For Cvfem_Solvent_Momentum ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Open_Flow ([Toggle = *toggle*] | [Pressure = *pressure*] | [Total_Pressure = *total_pressure*])

Summary

CVFEM solvent momentum open flow flux BC

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
pressure	real	0
total_pressure	real	0

Symmetry_Flow

Syntax

Bc No_Coverage_Flux For Cvfem_Solvent_Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Symmetry_Flow [Toggle = *toggle*]

Summary

CVFEM solvent momentum symmetry flow flux BC

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–

Vector_String_Function

Syntax

Bc No_Coverage_Flux For Cvfem_Solvent_Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Vector_String_Function ([Toggle = *toggle*] | F_X = *f_x* | F_Y = *f_y* | [F_Z = *f_z*])

Summary

Value from a string function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
f_x	“string”	–
f_y	“string”	–
f_z	“string”	–

Cvfem_Solvent_Momentum_X

Constant

Syntax

Bc No_Coverage_Flux For Cvfem_Solvent_Momentum_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | Value = *value*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
value	real	–

Exponential

Syntax

Bc No_Coverage_Flux For Cvfem_Solvent_Momentum_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc No_Coverage_Flux For Cvfem_Solvent_Momentum_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Bc No_Coverage_Flux For Cvfem_Solvent_Momentum_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc No_Coverage_Flux For Cvfem_Solvent_Momentum_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc No_Coverage_Flux For Cvfem_Solvent_Momentum_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f	"string"	–

User_Field

Syntax

Bc No_Coverage_Flux For Cvfem_Solvent_Momentum_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc No_Coverage_Flux For Cvfem_Solvent_Momentum_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc No_Coverage_Flux For Cvfem_Solvent_Momentum_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

Cvfem_Solvent_Momentum_Y

Constant

Syntax

Bc No_Coverage_Flux For Cvfem_Solvent_Momentum_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | Value = *value*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
value	real	–

Exponential

Syntax

Bc No_Coverage_Flux For Cvfem_Solvent_Momentum_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc No_Coverage_Flux For Cvfem_Solvent_Momentum_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Bc No_Coverage_Flux For Cvfem_Solvent_Momentum_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc No_Coverage_Flux For Cvfem_Solvent_Momentum_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc No_Coverage_Flux For Cvfem_Solvent_Momentum_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f	"string"	–

User_Field

Syntax

Bc No_Coverage_Flux For Cvfem_Solvent_Momentum_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc No_Coverage_Flux For Cvfem_Solvent_Momentum_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc No_Coverage_Flux For Cvfem_Solvent_Momentum_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

Cvfem_Solvent_Momentum_Z

Constant

Syntax

Bc No_Coverage_Flux For Cvfem_Solvent_Momentum_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | Value = *value*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
value	real	–

Exponential

Syntax

Bc No_Coverage_Flux For Cvfem_Solvent_Momentum_Z ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc No_Coverage_Flux For Cvfem_Solvent_Momentum_Z ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Bc No_Coverage_Flux For Cvfem_Solvent_Momentum_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc No_Coverage_Flux For Cvfem_Solvent_Momentum_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc No_Coverage_Flux For Cvfem_Solvent_Momentum_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f	"string"	–

User_Field

Syntax

Bc No_Coverage_Flux For Cvfem_Solvent_Momentum_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc No_Coverage_Flux For Cvfem_Solvent_Momentum_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc No_Coverage_Flux For Cvfem_Solvent_Momentum_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

Cvfem_Specific_Dissipation_Rate

Constant

Syntax

Bc No_Coverage_Flux For Cvfem_Specific_Dissipation_Rate ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | Value = *value*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
value	real	–

Exponential

Syntax

Bc No_Coverage_Flux For Cvfem_Specific_Dissipation_Rate ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc No_Coverage_Flux For Cvfem_Specific_Dissipation_Rate ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
global_name	"string"	–
component	integer	0

Mass_Flux

Syntax

Bc No_Coverage_Flux For Cvfem_Specific_Dissipation_Rate ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Mass_Flux ([Toggle = *toggle*] | Total_Mdot = *total_mdot* | [Inflow_Phi = *inflow_phi*])

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
total_mdot	real	–
inflow_phi	real	0

Nd_Table

Syntax

Bc No_Coverage_Flux For Cvfem_Specific_Dissipation_Rate ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Open_Adv_Flow

Syntax

Bc No_Coverage_Flux For Cvfem_Specific_Dissipation_Rate ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Open_Adv_Flow ([Toggle = *toggle*] | [Pressure = *pressure*] | [Total_Pressure = *total_pressure*] | [Far_Field_Entrainment_Value = *far_field_entrainment_value*])

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
pressure	real	0
total_pressure	real	0
far_field_entrainment_value	real	0

Polynomial

Syntax

Bc No_Coverage_Flux For Cvfem_Specific_Dissipation_Rate ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
variable	“string”	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc No_Coverage_Flux For Cvfem_Specific_Dissipation_Rate ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f	"string"	–

User_Field

Syntax

Bc No_Coverage_Flux For Cvfem_Specific_Dissipation_Rate ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc No_Coverage_Flux For Cvfem_Specific_Dissipation_Rate ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc No_Coverage_Flux For Cvfem_Specific_Dissipation_Rate ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

Cvfem_Temperature_Edge_Gradient_Projection

Constant

Syntax

Bc No_Coverage_Flux For Cvfem_Temperature_Edge_Gradient_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | Value = *value*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
value	real	–

Exponential

Syntax

Bc No_Coverage_Flux For Cvfem_Temperature_Edge_Gradient_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
variable	“string”	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc No_Coverage_Flux For Cvfem_Temperature_Edge_Gradient_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Bc No_Coverage_Flux For Cvfem_Temperature_Edge_Gradient_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc No_Coverage_Flux For Cvfem_Temperature_Edge_Gradient_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc No_Coverage_Flux For Cvfem_Temperature_Edge_Gradient_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f	"string"	–

Simple_Interp

Syntax

Bc No_Coverage_Flux For Cvfem_Temperature_Edge_Gradient_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Simple_Interp [Toggle = *toggle*]

Summary

CVFEM edge-based projection simple interpolation flux BC

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–

User_Field

Syntax

Bc No_Coverage_Flux For Cvfem_Temperature_Edge_Gradient_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
scaling	real	1
global_var	“string”	–
time_function	“string”	–

User_Function

Syntax

Bc No_Coverage_Flux For Cvfem_Temperature_Edge_Gradient_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc No_Coverage_Flux For Cvfem_Temperature_Edge_Gradient_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

Cvfem_Tke_Edge_Gradient_Projection

Constant

Syntax

Bc No_Coverage_Flux For Cvfem_Tke_Edge_Gradient_Projection ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | Value = *value*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
value	real	–

Exponential

Syntax

Bc No_Coverage_Flux For Cvfem_Tke_Edge_Gradient_Projection ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
variable	“string”	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc No_Coverage_Flux For Cvfem_Tke_Edge_Gradient_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Bc No_Coverage_Flux For Cvfem_Tke_Edge_Gradient_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc No_Coverage_Flux For Cvfem_Tke_Edge_Gradient_Projection ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc No_Coverage_Flux For Cvfem_Tke_Edge_Gradient_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f	"string"	–

Simple_Interp

Syntax

Bc No_Coverage_Flux For Cvfem_Tke_Edge_Gradient_Projection ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Simple_Interp [Toggle = *toggle*]

Summary

CVFEM edge-based projection simple interpolation flux BC

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–

User_Field

Syntax

Bc No_Coverage_Flux For Cvfem_Tke_Edge_Gradient_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
scaling	real	1
global_var	“string”	–
time_function	“string”	–

User_Function

Syntax

Bc No_Coverage_Flux For Cvfem_Tke_Edge_Gradient_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc No_Coverage_Flux For Cvfem_Tke_Edge_Gradient_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

Cvfem_Turbulence_Dissipation_Rate

Constant

Syntax

Bc No_Coverage_Flux For Cvfem_Turbulence_Dissipation_Rate ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | Value = *value*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
value	real	–

Exponential

Syntax

Bc No_Coverage_Flux For Cvfem_Turbulence_Dissipation_Rate ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Free_Open_Flow

Syntax

Bc No_Coverage_Flux For Cvfem_Turbulence_Dissipation_Rate ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Free_Open_Flow [Toggle = *toggle*]

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–

Global

Syntax

Bc No_Coverage_Flux For Cvfem_Turbulence_Dissipation_Rate ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
global_name	“string”	–
component	integer	0

Mass_Flux

Syntax

Bc No_Coverage_Flux For Cvfem_Turbulence_Dissipation_Rate ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Mass_Flux ([Toggle = *toggle*] | Total_Mdot = *total_mdot* | [Inflow_Phi = *inflow_phi*])

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
total_mdot	real	–
inflow_phi	real	0

Nd_Table

Syntax

Bc No_Coverage_Flux For Cvfem_Turbulence_Dissipation_Rate ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Open_Adv_Flow

Syntax

Bc No_Coverage_Flux For Cvfem_Turbulence_Dissipation_Rate ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Open_Adv_Flow ([Toggle = *toggle*] | [Pressure = *pressure*] | [Total_Pressure = *total_pressure*] | [Far_Field_Entrainment_Value = *far_field_entrainment_value*])

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
pressure	real	0
total_pressure	real	0
far_field_entrainment_value	real	0

Polynomial

Syntax

Bc No_Coverage_Flux For Cvfem_Turbulence_Dissipation_Rate ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc No_Coverage_Flux For Cvfem_Turbulence_Dissipation_Rate ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f	"string"	–

User_Field

Syntax

Bc No_Coverage_Flux For Cvfem_Turbulence_Dissipation_Rate ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc No_Coverage_Flux For Cvfem_Turbulence_Dissipation_Rate ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc No_Coverage_Flux For Cvfem_Turbulence_Dissipation_Rate ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

Cvfem_Turbulent_Kinetic_Energy

Constant

Syntax

Bc No_Coverage_Flux For Cvfem_Turbulent_Kinetic_Energy ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | Value = *value*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
value	real	–

Exponential

Syntax

Bc No_Coverage_Flux For Cvfem_Turbulent_Kinetic_Energy ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Free_Open_Flow

Syntax

Bc No_Coverage_Flux For Cvfem_Turbulent_Kinetic_Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Free_Open_Flow [Toggle = *toggle*]

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–

Global

Syntax

Bc No_Coverage_Flux For Cvfem_Turbulent_Kinetic_Energy ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
global_name	“string”	–
component	integer	0

Mass_Flux

Syntax

Bc No_Coverage_Flux For Cvfem_Turbulent_Kinetic_Energy ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Mass_Flux ([Toggle = *toggle*] | Total_Mdot = *total_mdot* | [Inflow_Phi = *inflow_phi*])

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
total_mdot	real	–
inflow_phi	real	0

Nd_Table

Syntax

Bc No_Coverage_Flux For Cvfem_Turbulent_Kinetic_Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Open_Adv_Flow

Syntax

Bc No_Coverage_Flux For Cvfem_Turbulent_Kinetic_Energy ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Open_Adv_Flow ([Toggle = *toggle*] | [Pressure = *pressure*] | [Total_Pressure = *total_pressure*] | [Far_Field_Entrainment_Value = *far_field_entrainment_value*])

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
pressure	real	0
total_pressure	real	0
far_field_entrainment_value	real	0

Polynomial

Syntax

Bc No_Coverage_Flux For Cvfem_Turbulent_Kinetic_Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc No_Coverage_Flux For Cvfem_Turbulent_Kinetic_Energy ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f	"string"	–

User_Field

Syntax

Bc No_Coverage_Flux For Cvfem_Turbulent_Kinetic_Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc No_Coverage_Flux For Cvfem_Turbulent_Kinetic_Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc No_Coverage_Flux For Cvfem_Turbulent_Kinetic_Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

Cvfem_Velocity_Edge_Gradient_Projection

Constant

Syntax

Bc No_Coverage_Flux For Cvfem_Velocity_Edge_Gradient_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | Value = *value*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
value	real	–

Exponential

Syntax

Bc No_Coverage_Flux For Cvfem_Velocity_Edge_Gradient_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc No_Coverage_Flux For Cvfem_Velocity_Edge_Gradient_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
global_name	“string”	–
component	integer	0

Nd_Table

Syntax

Bc No_Coverage_Flux For Cvfem_Velocity_Edge_Gradient_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc No_Coverage_Flux For Cvfem_Velocity_Edge_Gradient_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc No_Coverage_Flux For Cvfem_Velocity_Edge_Gradient_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f	"string"	–

User_Field

Syntax

Bc No_Coverage_Flux For Cvfem_Velocity_Edge_Gradient_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc No_Coverage_Flux For Cvfem_Velocity_Edge_Gradient_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc No_Coverage_Flux For Cvfem_Velocity_Edge_Gradient_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

Cvfem_Velocity_Pressure_Projection

Constant

Syntax

Bc No_Coverage_Flux For Cvfem_Velocity_Pressure_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | [Valuex = *valuex*] | [Valuey = *valuey*] | [Valuez = *valuez*])

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
valuex	real	0
valuey	real	0
valuez	real	0

Simple_Dp_Interp

Syntax

Bc No_Coverage_Flux For Cvfem_Velocity_Pressure_Projection ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Simple_Dp_Interp [Toggle = *toggle*]

Summary

Simple DP interpolation model for CVFEM velocity pressure projection flux

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–

Vector_String_Function

Syntax

Bc No_Coverage_Flux For Cvfem_Velocity_Pressure_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Vector_String_Function ([Toggle = *toggle*] | F_X = *f_x* | F_Y = *f_y* | [F_Z = *f_z*])

Summary

Value from a string function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
f_x	“string”	–
f_y	“string”	–
f_z	“string”	–

Cvfem_Velocity_Pressure_Projection_X

Constant

Syntax

Bc No_Coverage_Flux For Cvfem_Velocity_Pressure_Projection_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | Value = *value*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
value	real	–

Exponential

Syntax

Bc No_Coverage_Flux For Cvfem_Velocity_Pressure_Projection_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc No_Coverage_Flux For Cvfem_Velocity_Pressure_Projection_X ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
global_name	“string”	–
component	integer	0

Nd_Table

Syntax

Bc No_Coverage_Flux For Cvfem_Velocity_Pressure_Projection_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc No_Coverage_Flux For Cvfem_Velocity_Pressure_Projection_X ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc No_Coverage_Flux For Cvfem_Velocity_Pressure_Projection_X ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f	"string"	–

User_Field

Syntax

Bc No_Coverage_Flux For Cvfem_Velocity_Pressure_Projection_X ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc No_Coverage_Flux For Cvfem_Velocity_Pressure_Projection_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc No_Coverage_Flux For Cvfem_Velocity_Pressure_Projection_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

Cvfem_Velocity_Pressure_Projection_Y

Constant

Syntax

Bc No_Coverage_Flux For Cvfem_Velocity_Pressure_Projection_Y ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | Value = *value*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
value	real	–

Exponential

Syntax

Bc No_Coverage_Flux For Cvfem_Velocity_Pressure_Projection_Y ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc No_Coverage_Flux For Cvfem_Velocity_Pressure_Projection_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
global_name	“string”	–
component	integer	0

Nd_Table

Syntax

Bc No_Coverage_Flux For Cvfem_Velocity_Pressure_Projection_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc No_Coverage_Flux For Cvfem_Velocity_Pressure_Projection_Y ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc No_Coverage_Flux For Cvfem_Velocity_Pressure_Projection_Y ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f	"string"	–

User_Field

Syntax

Bc No_Coverage_Flux For Cvfem_Velocity_Pressure_Projection_Y ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc No_Coverage_Flux For Cvfem_Velocity_Pressure_Projection_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc No_Coverage_Flux For Cvfem_Velocity_Pressure_Projection_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

Cvfem_Velocity_Pressure_Projection_Z

Constant

Syntax

Bc No_Coverage_Flux For Cvfem_Velocity_Pressure_Projection_Z ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | Value = *value*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
value	real	–

Exponential

Syntax

Bc No_Coverage_Flux For Cvfem_Velocity_Pressure_Projection_Z ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc No_Coverage_Flux For Cvfem_Velocity_Pressure_Projection_Z ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Bc No_Coverage_Flux For Cvfem_Velocity_Pressure_Projection_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc No_Coverage_Flux For Cvfem_Velocity_Pressure_Projection_Z ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc No_Coverage_Flux For Cvfem_Velocity_Pressure_Projection_Z ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f	"string"	–

User_Field

Syntax

Bc No_Coverage_Flux For Cvfem_Velocity_Pressure_Projection_Z ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc No_Coverage_Flux For Cvfem_Velocity_Pressure_Projection_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc No_Coverage_Flux For Cvfem_Velocity_Pressure_Projection_Z ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

Darcy_Momentum

Constant

Syntax

Bc No_Coverage_Flux For Darcy_Momentum ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | [Valuex = *valuex*] | [Valuey = *valuey*] | [Valuez = *valuez*])

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
valuex	real	0
valuey	real	0
valuez	real	0

Open_Flow

Syntax

Bc No_Coverage_Flux For Darcy_Momentum ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Open_Flow ([Toggle = *toggle*] | [Pressure = *pressure*] | [Total_Pressure = *total_pressure*])

Summary

Open flow momentum flux

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
pressure	real	0
total_pressure	real	0

Pressure

Syntax

Bc No_Coverage_Flux For Darcy_Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Pressure ([Toggle = *toggle*] | [User_Field_Name = *user_field_name*] | [Use_Bulk_Node = *use_bulk_node*] | [P = *p*] | [C_T = *c_t*] | [C_X = *c_x*] | [C_Y = *c_y*] | [C_Z = *c_z*] | [C_Cos_W = *c_cos_w*] | [C_Sin_W = *c_sin_w*] | [C1_Cos = *c1_cos*] | [C1_Sin = *c1_sin*])

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
user_field_name	“string”	–
use_bulk_node	“string”	–
p	real	0
c_t	real	0
c_x	real	0
c_y	real	0
c_z	real	0
c_cos_w	real	0
c_sin_w	real	0
c1_cos	real	0
c1_sin	real	0

Rt_Pressure

Syntax

Bc No_Coverage_Flux For Darcy_Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Rt_Pressure ([Toggle = *toggle*] | P = *p* | [C_T = *c_t*] | [C_X = *c_x*] | [C_Y = *c_y*] | [C_Z = *c_z*])

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
p	real	–
c_t	real	0
c_x	real	0
c_y	real	0
c_z	real	0

Simple_Interp

Syntax

Bc No_Coverage_Flux For Darcy_Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Simple_Interp [Toggle = *toggle*]

Summary

Darcy momentum simple interpolation flux BC

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–

Slip

Syntax

Bc No_Coverage_Flux For Darcy_Momentum ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Slip ([Toggle = *toggle*] | [Vs_X = *vs_x*] | [Vs_Y = *vs_y*] | [Vs_Z = *vs_z*] | K = *k*)

Summary

Darcy slip momentum flux

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
vs_x	real	0
vs_y	real	0
vs_z	real	0
k	real	–

Vector_String_Function

Syntax

Bc No_Coverage_Flux For Darcy_Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Vector_String_Function ([Toggle = *toggle*] | F_X = *f_x* | F_Y = *f_y* | [F_Z = *f_z*])

Summary

Value from a string function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
f_x	“string”	–
f_y	“string”	–
f_z	“string”	–

Darcy_Momentum_X

Constant

Syntax

Bc No_Coverage_Flux For Darcy_Momentum_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | Value = *value*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
value	real	–

Exponential

Syntax

Bc No_Coverage_Flux For Darcy_Momentum_X ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc No_Coverage_Flux For Darcy_Momentum_X ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Bc No_Coverage_Flux For Darcy_Momentum_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc No_Coverage_Flux For Darcy_Momentum_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc No_Coverage_Flux For Darcy_Momentum_X ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f	"string"	–

User_Field

Syntax

Bc No_Coverage_Flux For Darcy_Momentum_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
scaling	real	1
global_var	“string”	–
time_function	“string”	–

User_Function

Syntax

Bc No_Coverage_Flux For Darcy_Momentum_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc No_Coverage_Flux For Darcy_Momentum_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

Darcy_Momentum_Y

Constant

Syntax

Bc No_Coverage_Flux For Darcy_Momentum_Y ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | Value = *value*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
value	real	–

Exponential

Syntax

Bc No_Coverage_Flux For Darcy_Momentum_Y ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc No_Coverage_Flux For Darcy_Momentum_Y ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Bc No_Coverage_Flux For Darcy_Momentum_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc No_Coverage_Flux For Darcy_Momentum_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc No_Coverage_Flux For Darcy_Momentum_Y ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f	"string"	–

User_Field

Syntax

Bc No_Coverage_Flux For Darcy_Momentum_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
scaling	real	1
global_var	“string”	–
time_function	“string”	–

User_Function

Syntax

Bc No_Coverage_Flux For Darcy_Momentum_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc No_Coverage_Flux For Darcy_Momentum_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

Darcy_Momentum_Z

Constant

Syntax

Bc No_Coverage_Flux For Darcy_Momentum_Z ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | Value = *value*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
value	real	–

Exponential

Syntax

Bc No_Coverage_Flux For Darcy_Momentum_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc No_Coverage_Flux For Darcy_Momentum_Z ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Bc No_Coverage_Flux For Darcy_Momentum_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc No_Coverage_Flux For Darcy_Momentum_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc No_Coverage_Flux For Darcy_Momentum_Z ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f	"string"	–

User_Field

Syntax

Bc No_Coverage_Flux For Darcy_Momentum_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc No_Coverage_Flux For Darcy_Momentum_Z ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc No_Coverage_Flux For Darcy_Momentum_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

Div_Projection

Constant

Syntax

Bc No_Coverage_Flux For Div_Projection ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | [Valuex = *valuex*] | [Valuey = *valuey*] | [Valuez = *valuez*])

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
valuex	real	0
valuey	real	0
valuez	real	0

Vector_String_Function

Syntax

Bc No_Coverage_Flux For Div_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Vector_String_Function ([Toggle = *toggle*] | F_X = *f_x* | F_Y = *f_y* | [F_Z = *f_z*])

Summary

Value from a string function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f_x	"string"	–
f_y	"string"	–
f_z	"string"	–

Div_Projection_X

Constant

Syntax

Bc No_Coverage_Flux For Div_Projection_X ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | Value = *value*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
value	real	–

Exponential

Syntax

Bc No_Coverage_Flux For Div_Projection_X ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc No_Coverage_Flux For Div_Projection_X ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Bc No_Coverage_Flux For Div_Projection_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc No_Coverage_Flux For Div_Projection_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc No_Coverage_Flux For Div_Projection_X ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f	"string"	–

User_Field

Syntax

Bc No_Coverage_Flux For Div_Projection_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc No_Coverage_Flux For Div_Projection_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc No_Coverage_Flux For Div_Projection_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

Div_Projection_Y

Constant

Syntax

Bc No_Coverage_Flux For Div_Projection_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | Value = *value*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
value	real	–

Exponential

Syntax

Bc No_Coverage_Flux For Div_Projection_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc No_Coverage_Flux For Div_Projection_Y ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Bc No_Coverage_Flux For Div_Projection_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc No_Coverage_Flux For Div_Projection_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc No_Coverage_Flux For Div_Projection_Y ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f	"string"	–

User_Field

Syntax

Bc No_Coverage_Flux For Div_Projection_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
scaling	real	1
global_var	“string”	–
time_function	“string”	–

User_Function

Syntax

Bc No_Coverage_Flux For Div_Projection_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc No_Coverage_Flux For Div_Projection_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

Div_Projection_Z

Constant

Syntax

Bc No_Coverage_Flux For Div_Projection_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | Value = *value*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
value	real	–

Exponential

Syntax

Bc No_Coverage_Flux For Div_Projection_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc No_Coverage_Flux For Div_Projection_Z ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Bc No_Coverage_Flux For Div_Projection_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc No_Coverage_Flux For Div_Projection_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc No_Coverage_Flux For Div_Projection_Z ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f	"string"	–

User_Field

Syntax

Bc No_Coverage_Flux For Div_Projection_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc No_Coverage_Flux For Div_Projection_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc No_Coverage_Flux For Div_Projection_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

Energy

Calore_User_Sub

Syntax

Bc No_Coverage_Flux For Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Calore_User_Sub ([Power_Output = *power_output*] | [Flux_Output = *flux_output*] | [Toggle = *toggle*] | Name = *name* | Type = *type* | [Multiplier = *multiplier*] | [Material_Data_Block = *material_data_block*] | [Data = *data*] | [Scaling_Field = *scaling_field*])

Summary

Values from a Calore user subroutine

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output

If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Flux_Output

If specified, the average flux of the given BC/Src is calculated and stored into the global variable name specified as the *flux_output* argument.

Toggle

If given, specifies which *toggle block* controls whether this BC is

active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
power_output	“string”	–
flux_output	“string”	–
toggle	“string”	–
name	“string”	–
type	“string”	–
multiplier	real	1
material_data_block	“string”	–
data	“string”	–
scaling_field	“string”	–

Cht_Flux

Syntax

Bc No_Coverage_Flux For Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Cht_Flux ([Power_Output = *power_output*] | [Flux_Output = *flux_output*] | [Toggle = *toggle*] | [Too_Field = *Too_Field*] | [H_Field = *H_Field*])

Summary

Applies H(T-Too) flux for conjugate heat transfer boundary. Optional parameters are “Too_Field” and “H_Field”, which default to “nodal_cht_Too” and “nodal_cht_H” respectively.

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output

If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Flux_Output

If specified, the average flux of the given BC/Src is calculated and stored into the global variable name specified as the *flux_output* argument.

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
power_output	“string”	–
flux_output	“string”	–
toggle	“string”	–
Too_Field	“string”	nodal_cht_Too
H_Field	“string”	nodal_cht_H

Cht_Robin

Syntax

Bc No_Coverage_Flux For Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Cht_Robin ([Power_Output = *power_output*] | [Flux_Output = *flux_output*] | [Toggle = *toggle*])

Summary

Applies a Robin-style BC of the form: J dot n + h * (T-Tref).

Description

J, h, and Tref should be specified using the material properties BC_HEAT_FLUX, HEAT_TRANSFER_COEFFICIENT, and BC_REFERENCE_TEMPERATURE in a material that is applied to the surface.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output

If specified, the total (integrated) power of the given BC/Src is

calculated and stored into the global variable name specified as the *power_output* argument.

Flux_Output

If specified, the average flux of the given BC/Src is calculated and stored into the global variable name specified as the *flux_output* argument.

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
power_output	"string"	–
flux_output	"string"	–
toggle	"string"	–

Constant

Syntax

Bc No_Coverage_Flux For Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Power_Output = *power_output*] | [Flux_Output = *flux_output*] | [Toggle = *toggle*] | *{flux | value}* = *flux*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output

If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Flux_Output

If specified, the average flux of the given BC/Src is calculated and stored into the global variable name specified as the *flux_output* argument.

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
power_output	"string"	–
flux_output	"string"	–
toggle	"string"	–
flux	real	–

Distribution_Factor

Syntax

Bc No_Coverage_Flux For Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Distribution_Factor ([Power_Output = *power_output*] | [Flux_Output = *flux_output*] | [Toggle = *toggle*] | [Attribute_Name = *attribute_name*] | [Multiplier = *multiplier*])

Summary

Applies a flux equal to a distribution factor defined in the input mesh with name ATTRIBUTE_NAME. F = MULTIPLIER x ATTRIBUTE_NAME[node]

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output

If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Flux_Output

If specified, the average flux of the given BC/Src is calculated and stored into the global variable name specified as the *flux_output* argument.

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
power_output	"string"	–
flux_output	"string"	–
toggle	"string"	–
attribute_name	"string"	DISTRIBUTION_FACTOR
multiplier	real	1

Enclosure_Radiation

Syntax

Bc No_Coverage_Flux For Energy ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Enclosure_Radiation ([Power_Output = *power_output*] | [Flux_Output = *flux_output*] | [Toggle = *toggle*] | [Multiplier = *multiplier*] | Enclosure = *enclosure*)

Summary

This term imposes an energy flux due to enclosure radiation as defined in *Enclosure Radiation*. The irradiation is provided by the Chapparal radiosity solution

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output

If specified, the total (integrated) power of the given BC/Src is

calculated and stored into the global variable name specified as the *power_output* argument.

Flux_Output

If specified, the average flux of the given BC/Src is calculated and stored into the global variable name specified as the *flux_output* argument.

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
power_output	"string"	–
flux_output	"string"	–
toggle	"string"	–
multiplier	real	1
enclosure	"string"	–

Exponential

Syntax

Bc No_Coverage_Flux For Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Power_Output = *power_output*] | [Flux_Output = *flux_output*] | [Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

$$F = C + Me^{kX}$$

where X is the variable, C is the constant, M is the multiplier, and k is exponent. C and M are optional and default to 0 and 1 respectively.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output
If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Flux_Output
If specified, the average flux of the given BC/Src is calculated and stored into the global variable name specified as the *flux_output* argument.

Toggle
If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
power_output	"string"	–
flux_output	"string"	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Exponential_Vapor_Cooling

Syntax

Bc No_Coverage_Flux For Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential_Vapor_Cooling ([Power_Output = *power_output*] | [Flux_Output = *flux_output*] | [Toggle = *toggle*] | Tboil = *tboil* | L = *l* | Pa = *pa* | Mw = *mw* | Alpha = *alpha* | [Tl = *tl*] | [Th = *th*])

Summary

Exponential vapor cooling energy flux

Description

Flux is given by:

$$F = P_a L\alpha \exp\left(\frac{L}{R}\left(\frac{1}{T_{boil}} - \frac{1}{T}\right)\right)\sqrt{\frac{1}{2\pi RT}}$$

where R is the specific gas constant ($\bar{R}$/MW). For T less than T_L, zero flux is applied. For T between T_L and T_H, a 3rd order smoothing polynomial is used to provide continuity of values and derivatives between 0 and the expression given above. For T greater than T_H, the above expression is used. If not provided, T_L and T_H are set to 10% below and above T_{Boil}, respectively.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output

If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Flux_Output

If specified, the average flux of the given BC/Src is calculated and stored into the global variable name specified as the *flux_output* argument.

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
power_output	"string"	–
flux_output	"string"	–
toggle	"string"	–
tboil	real	–
l	real	–
pa	real	–
mw	real	–
alpha	real	–
tl	real	-1
th	real	-1

Face_Field_Scalar

Syntax

Bc No_Coverage_Flux For Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Face_Field_Scalar ([Power_Output = *power_output*] | [Flux_Output = *flux_output*] | [Toggle = *toggle*] | Name = *name* | [Multiplier = *multiplier*] | [Advective_Bar = *advective_bar*] | [Command_Block_Name = *command_block_name*])

Summary

FACE or EDGE Field energy flux

Description

Flux is given by:

$$M \cdot f$$

M is given by MULTIPLIER and f is a face or edge field with name NAME.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output

If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Flux_Output

If specified, the average flux of the given BC/Src is calculated and stored into the global variable name specified as the *flux_output* argument.

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
power_output	“string”	–
flux_output	“string”	–
toggle	“string”	–
name	“string”	–
multiplier	real	1
advective_bar	“string”	–
command_block_name	“string”	–

Fortran

Syntax

Bc No_Coverage_Flux For Energy ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Fortran ([Power_Output = *power_output*] | [Flux_Output = *flux_output*] | [Toggle = *toggle*] | [Multiplier = *multiplier*] | Sub_Name = *sub_name* | [Real_Data = *real_data*] | [Int_Data = *int_data*] | [Resource_Name = *resource_name*] | [Data = *data*])

Summary

Fortran subroutine energy flux

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output

If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Flux_Output

If specified, the average flux of the given BC/Src is calculated and stored into the global variable name specified as the *flux_output* argument.

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
power_output	"string"	–
flux_output	"string"	–
toggle	"string"	–
multiplier	real	1
sub_name	"string"	–
real_data	"string"	–
int_data	"string"	–
resource_name	"string"	–
data	"string"	–

Gaussian_Power_Weld

Syntax

Bc No_Coverage_Flux For Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Gaussian_Power_Weld ([Power_Output = *power_output*] | [Flux_Output = *flux_output*] | [Toggle = *toggle*] | R = *r* | [Clip_Radius = *clip_radius*] | [Efficiency = *efficiency*] | R_Eff = *R_eff* | Origin_And_Normal_Function = *origin_and_normal_function* | [Old_Temperature = *old_temperature*] | Path_Function = *path_function* | [Deposition = *deposition*] | [Integrated = *integrated*] | [Clip = *clip*])

Summary

Energy flux due to an incident Gaussian laser beam

Description

The coordinates of the center of the beam are provided by the X, Y, and Z components of the provided origin_and_normal_function. The direction of the beam is provided by the NX, NY, and NZ components of the origin_and_normal_function. The applied flux is given by:

$$\frac{(-\boldsymbol{n})\cdot\boldsymbol{d}}{|\boldsymbol{d}|}\frac{\alpha e R_{eff} P}{\pi R^2(1-e^{-R_{eff}})}e^{-R_{eff}\frac{r_p^2}{R^2}}$$

P is the laser power defined for the material, R is the beam radius, R_{eff} is the effective beam radius, α is the laser absorptivity defined for the material, e is the EFFICIENCY, and r_p is the shortest distance between a point on the surface and the line defined by the beam origin and normal. The term $-\boldsymbol{n}\cdot\boldsymbol{d}$ accounts for the fact that the surface may not be orthogonal to the laser. Points with values of r_p > CLIP_RADIUS will receive zero flux. If ACTIVATION is set to true, points whose temperature exceed ACTIVATION_TEMPERATURE will be flagged as activated for use with the ACTIVATION USER FUNCTION thermal conductivity model. If INTEGRATED is true, this flux will be integrated between the previous time step and the current one and the average flux applied.

See the *Directed Energy* section for the appropriate syntax to use this flux in the directed energy block.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output

If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Flux_Output

If specified, the average flux of the given BC/Src is calculated and

stored into the global variable name specified as the *flux_output* argument.

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
power_output	"string"	–
flux_output	"string"	–
toggle	"string"	–
r	real	–
clip_radius	real	0
efficiency	real	1
R_eff	real	–
origin_and_normal_function	"string"	–
old_temperature	"string"	TRUE
path_function	"string"	–
deposition	"string"	FALSE
integrated	"string"	FALSE
clip	"string"	TRUE

Gaussian_Ray_Trace

Syntax

Bc No_Coverage_Flux For Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Gaussian_Ray_Trace ([Power_Output = *power_output*] | [Flux_Output = *flux_output*] | [Toggle = *toggle*] | Path_Function = *path_function* | Dir = *dir* | Power = *power* | Type = *type* | [N_Rays = *n_rays*] | [N_R = *n_r*] | [N_Theta_Initial = *n_theta_initial*] | [N_Intersections = *n_intersections*] | [Weight_Ratio = *weight_ratio*] | [Id = *id*] | [Absorption_Model = *absorption_model*] | [Parallel_Type = *parallel_type*] | R = *r* | R_Eff = *r_eff*)

Summary

Ray tracing implementation of a Gaussian Beam energy flux

Description

r - Laser Gaussian standard deviation.

r_eff - Laser exponential factor. Gaussian is given by $EXP\left(\frac{-r_eff*D^2}{r^2}\right)$, where D is the distance between a point and the laser center.

type - Either radial discretization or Monte Carlo. Determines how beam will be discretized into rays. Radial discretization is deterministic and breaks beams into approximately equal area segments radially. Each segment generates one ray. The power of each ray varies based on which segment it is generated from. Monte Carlo is stochastic and will generate a user-specified number of rays distributed spatially according to the laser's shape function, with each ray having equal power of power/n_rays.

n_r - Number of segments in radial direction for radial discretization.

n_theta_initial - Number of segments in theta direction in the smallest radial segment. The number of theta segments will grow as radial direction increases so segments have approximately the same area. Total number of rays will be n_r*n_r*n_theta_initial.

n_rays - Number of rays to generate for Monte Carlo discretization.

path_function - Function giving x, y, z coordinates of laser center as function of time.

dir - Function giving x,y,z components of direction that rays will travel initially from laser center defined by path_function.

power - Function giving total laser power.

n_intersections - Maximum number of times to intersect a ray with mesh elements due to reflections (default: 10).

weight_ratio - Fraction of initial ray power below which rays will be considered spent and no longer be traced (default: 0.001).

absorption_model - Either Specular, to use a surface emissivity value to determine fraction of energy absorbed and reflected with each ray intersection or Fresnel to use Fresnel model dependent on ray angle of incidence.

parallel_type - Either ray_parallel, to divide rays between processors with a globally shared mesh part or scene_parallel, to divide mesh part between processors with a globally shared set of rays.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output

If specified, the total (integrated) power of the given BC/Src is

calculated and stored into the global variable name specified as the *power_output* argument.

Flux_Output

If specified, the average flux of the given BC/Src is calculated and stored into the global variable name specified as the *flux_output* argument.

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
power_output	"string"	–
flux_output	"string"	–
toggle	"string"	–
path_function	"string"	–
dir	"string"	–
power	"string"	–
type	"string"	–
n_rays	integer	0
n_r	integer	0
n_theta_initial	integer	0
n_intersections	integer	10
weight_ratio	real	0.001
id	"string"	–
absorption_model	"string"	SPECULAR
parallel_type	"string"	RAY_PARALLEL
r	real	–
r_eff	real	–

Generalized_Nat_Conv

Syntax

Bc No_Coverage_Flux For Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Generalized_Nat_Conv ([Power_Output = *power_output*] | [Flux_Output = *flux_output*] | [Toggle = *toggle*] | [Multiplier = *multiplier*] | [H = *h*] | [T_Ref = *t_ref*])

Summary

Generalized natural convection energy flux

Description

Flux is given by:

$$F = Mh\left(T - T_{ref}\right)$$

M is given by MULTIPLIER, T_{ref} and h are taken from the BC REFERENCE TEMPERATURE and HEAT TRANSFER COEFFICIENT expressions.

The values for h and T_{ref} can also be supplied in-line as either constants or string functions.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output

If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Flux_Output

If specified, the average flux of the given BC/Src is calculated and stored into the global variable name specified as the *flux_output* argument.

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
power_output	"string"	–
flux_output	"string"	–
toggle	"string"	–
multiplier	real	1
h	"string"	–
t_ref	"string"	–

Generalized_Rad

Syntax

Bc No_Coverage_Flux For Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Generalized_Rad ([Power_Output = *power_output*] | [Flux_Output = *flux_output*] | [Toggle = *toggle*] | [Multiplier = *multiplier*] | [T_Ref = *t_ref*])

Summary

Radiative energy flux, with values for emissivity, form factor and irradiation taken from expressions

Description

Flux is:

$$M\epsilon f\left(\sigma T^4 - I\right)$$

M is MULTIPLIER. σ is the Stefan-Boltzmann constant. ϵ and F are expressions for emissivity and form factor. T is surface temperature. I is the expression for irradiation. If I does not already exist, it will be constructed as σT_{ref}^4, where T_{ref} is either provided by the BC Rad Reference Temperature expression or the inline T_{ref} parameter (constant or string function).

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output

If specified, the total (integrated) power of the given BC/Src is

calculated and stored into the global variable name specified as the *power_output* argument.

Flux_Output

If specified, the average flux of the given BC/Src is calculated and stored into the global variable name specified as the *flux_output* argument.

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
power_output	"string"	–
flux_output	"string"	–
toggle	"string"	–
multiplier	real	1
t_ref	"string"	–

Global

Syntax

Bc No_Coverage_Flux For Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Power_Output = *power_output*] | [Flux_Output = *flux_output*] | [Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output

If specified, the total (integrated) power of the given BC/Src is

calculated and stored into the global variable name specified as the *power_output* argument.

Flux_Output

If specified, the average flux of the given BC/Src is calculated and stored into the global variable name specified as the *flux_output* argument.

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
power_output	"string"	–
flux_output	"string"	–
toggle	"string"	–
global_name	"string"	–
component	integer	0

Kuntz

Syntax

Bc No_Coverage_Flux For Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Kuntz ([Power_Output = *power_output*] | [Flux_Output = *flux_output*] | [Toggle = *toggle*] | [Multiplier = *multiplier*] | File = *file* | Units = *units*)

Summary

Interpolated value from tabulated data for the energy flux. FILE provides the filename for a table which specifies a flux value given X, Y, and Z coordinates and a temperature. UNITS specifies the units the table is in - ENGLISH ($\frac{BTU}{sft^2}$) or METRIC($\frac{W}{m^2}$)

Description

The following options are present in most boundary conditions/sources. The

meaning of each option is as follows:

Power_Output
If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Flux_Output
If specified, the average flux of the given BC/Src is calculated and stored into the global variable name specified as the *flux_output* argument.

Toggle
If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
power_output	“string”	–
flux_output	“string”	–
toggle	“string”	–
multiplier	real	1
file	“string”	–
units	“string”	–

Lens

Syntax

Bc No_Coverage_Flux For Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Lens ([Power_Output = *power_output*] | [Flux_Output = *flux_output*] | [Toggle = *toggle*] | Src_X = *src_x* | Dir_X = *dir_x* | Vel_X = *vel_x* | Src_Y = *src_y* | Dir_Y = *dir_y* | Vel_Y = *vel_y* | [Src_Z = *src_z*] | [Dir_Z = *dir_z*] | [Vel_Z = *vel_z*] | Massflowrate = *massFlowRate* | T_Liq = *T_liq* | T_Solid = *T_solid* | [T_On = *t_on*] | [T_Off = *t_off*] | Masssourceradius = *massSourceRadius* | Particletemperature = *particleTemperature* | Particledensity = *particleDensity*)

Summary

Energy flux for modeling the LENS process

Description

This boundary condition imposes an energy flux due to an incident mass source representing a powder particle stream due to the LENS process. The source is at the coordinates

$$\boldsymbol{s} = \boldsymbol{x}_s + \boldsymbol{v}_s(t - T_ON)$$

where $\boldsymbol{x}_s$ is provided by the SRC vector coordinates and $\boldsymbol{v}_s$ is provided by the VEL vector coordinates. The source is directed from by the DIR vector (). The point-wise energy flux is computed using the equation below:

$$\frac{(-\boldsymbol{n}) \cdot \boldsymbol{d}}{|\boldsymbol{d}|} \left(\rho c_p T \frac{\dot{m}}{\rho_p \pi R^2} - \frac{\dot{m} c_p T_p}{\pi R^2} \right)$$

ρ, c_p, and T are the density, specific heat, and temperature of the surface. R, $\dot{m}$, ρ_p, and T_p are given by the massSourceRadius, massFlowRate, particleDensity, and particleTemperature parameters. The term $-\boldsymbol{n} \cdot \boldsymbol{d}$ accounts for the fact that the surface may not be orthogonal to the source. For points where the minimum distance between the point and the line defined by and is greater than R, the flux will be set to zero. Additionally, if T is less than (T_liq+T_solid)/2, the flux will be set to zero. For times less than T_ON or greater than T_OFF, the flux will also be set to zero.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output

If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Flux_Output

If specified, the average flux of the given BC/Src is calculated and stored into the global variable name specified as the *flux_output* argument.

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
power_output	“string”	–
flux_output	“string”	–
toggle	“string”	–
src_x	real	–
dir_x	real	–
vel_x	real	–
src_y	real	–
dir_y	real	–
vel_y	real	–
src_z	real	0
dir_z	real	0
vel_z	real	0
massFlowRate	real	–
T_liq	real	–
T_solid	real	–
t_on	real	0
t_off	real	1.798e+308
massSourceRadius	real	–
particleTemperature	real	–
particleDensity	real	–

Nat_Conv

Syntax

Bc No_Coverage_Flux For Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nat_Conv ([Power_Output = *power_output*] | [Flux_Output = *flux_output*] | [Toggle = *toggle*] | H = *h* | T_Ref = *t_ref*)

Summary

Convective energy flux defined with constants specified for heat transfer coefficient and reference temperature.

Description

Flux is given by:

$$F = h\left(T - T_{ref}\right)$$

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output

If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Flux_Output

If specified, the average flux of the given BC/Src is calculated and stored into the global variable name specified as the *flux_output* argument.

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
power_output	"string"	–
flux_output	"string"	–
toggle	"string"	–
h	real	–
t_ref	real	–

Nd_Table

Syntax

Bc No_Coverage_Flux For Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Power_Output = *power_output*] | [Flux_Output = *flux_output*] |

[Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table.

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output

If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Flux_Output

If specified, the average flux of the given BC/Src is calculated and stored into the global variable name specified as the *flux_output* argument.

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
power_output	“string”	–
flux_output	“string”	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Pid_Controlled

Syntax

Bc No_Coverage_Flux For Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Pid_Controlled ([Power_Output = *power_output*] | [Flux_Output = *flux_output*] | [Toggle = *toggle*] | Max_Output = *Max_Output* | [P = *p*] | [I = *i*] | [D = *d*] | [T_I = *t_i*] | [T_D = *t_d*] | [Band = *Band*] | [Filter_Tau = *Filter_Tau*] | [Filter_Number = *Filter_Number*] | Setpoint_Function_Name = *Setpoint_Function_Name* | Control_Variable = *Control_Variable* | [Enable_Clamping = *enable_clamping*])

Summary

This flux term imposes an energy flux using a PID controller

Description

The controller setpoint vs. time is defined in a user function, and the controller feedback point uses a global variable (e.g. a data probe or reduction postprocessor).

The derivative term uses a filtered error value, using a low pass filter whose time constant is specified by the user. Like a heater, this is only capable of providing a positive influx. Even if the controller output is negative, the applied energy source will be truncated at 0. Likewise, the applied energy source cannot exceed the specified max source.

The nominal scaled PID controller output (between 0 and 1) is

$$u(t) = K_p e(t) + K_i \int_0^t e(s)ds + K_d \frac{\partial e_f}{\partial t}$$

where it is important to note that the K_p coefficient does not scale the others unless you have set it using the Band argument. In some PID implementations, the K_i and K_d coefficients are internally multiplied by K_p so care must be taken to provide the correct coefficients. This output $u(t)$ which is from 0 to 1 is scaled by the specified Max_Output value to produce the applied energy source. This means you **should not** include the output magnitude scaling in your P, I, or D parameters. If this controller is meant to recreate a controller that commands power directly, one must normalize by the max power.

When a derivative gain is provided, the derivative action is calculated using a filtered error to avoid derivative kick. The `Filter_Tau` argument can be used to specify the first-order derivative filter time constant τ, with a default value of 1 otherwise. Alternatively, one can provide the `Filter_Number` as argument $\tau = \frac{1}{N}$ where

$$K_d \frac{N}{1 + N\frac{1}{s}} = K_d \frac{1}{\tau + \frac{1}{s}} = \frac{K_d s}{\tau s + 1}$$

The P, I, and D coefficients can be specified in a number of ways:

- Just give a 'Band' argument. This calculates $P = 1/Band$ and uses the defaults of $I = 0.01P$ and $D = 0.0$
- Give a 'Band' argument and non-defaults for I and/or D. This still multiplies whatever you give for I and D by P.
- Give a 'P' argument instead of 'Band'. NOTE: in this mode the I and D values are not scaled by P.

If you provide both P and Band, P is used, the Band argument is ignored, and I and D are not scaled.

Alternatively, the integral and/or derivative gains can be provided using the integral time T_i (`T_i` argument) and derivative time T_d (`T_d` argument) so that $I = P/T_i$ and $D = P * T_d$ (regardless if P was set via band arg or not).

If `ENABLE_CLAMPING` is set to true, a clamping anti-windup algorithm is applied to the integral action of the controller.

The default values for I and D are 0.01 and 0 if not provided. The setpoint, controller output, error, filtered error, and filtered error time derivative are all output to automatically created global variables for diagnostics and postprocessing.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output
: If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Flux_Output
: If specified, the average flux of the given BC/Src is calculated and stored into the global variable name specified as the *flux_output* argument.

Toggle
: If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
power_output	"string"	–
flux_output	"string"	–
toggle	"string"	–
Max_Output	real	–
p	real	0
i	real	0
d	real	0
t_i	real	0
t_d	real	0
Band	real	0
Filter_Tau	real	1
Filter_Number	real	1
Setpoint_Function_Name	"string"	–
Control_Variable	"string"	–
enable_clamping	integer	–

Pid_Controlled_Bidirectional

Syntax

Bc No_Coverage_Flux For Energy ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Pid_Controlled_Bidirectional ([Power_Output = *power_output*] | [Flux_Output = *flux_output*] | [Toggle = *toggle*] | Max_Cooling_Power = *Max_Cooling_Power* | Max_Heating_Power = *Max_Heating_Power* | [P = *p*] | [I = *i*] | [D = *d*] | [T_I = *t_i*] | [T_D = *t_d*] | [Band = *Band*] | [Filter_Tau = *Filter_Tau*] | [Filter_Number = *Filter_Number*] | Setpoint_Function_Name = *Setpoint_Function_Name* | Control_Variable = *Control_Variable* | [Enable_Clamping = *enable_clamping*])

Summary

This is similar to PID_Controlled, however, the flux is applied to cool and heat the volume instead of just heating it. Max_Cooling_Power and Max_Heating_Power should be positive values. See description for PID_CONTROLLED for more details

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output

If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Flux_Output

If specified, the average flux of the given BC/Src is calculated and stored into the global variable name specified as the *flux_output* argument.

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
power_output	"string"	–
flux_output	"string"	–
toggle	"string"	–
Max_Cooling_Power	real	–
Max_Heating_Power	real	–
p	real	0
i	real	0
d	real	0
t_i	real	0
t_d	real	0
Band	real	0
Filter_Tau	real	1
Filter_Number	real	1
Setpoint_Function_Name	"string"	–
Control_Variable	"string"	–
enable_clamping	integer	–

Pid_Controlled_Cooler

Syntax

Bc No_Coverage_Flux For Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Pid_Controlled_Cooler ([Power_Output = *power_output*] | [Flux_Output = *flux_output*] | [Toggle = *toggle*] | [P = *p*] | [I = *i*] | [D = *d*] | [T_I = *t_i*] | [T_D = *t_d*] | [Band = *Band*] | [Filter_Tau = *Filter_Tau*] | [Filter_Number = *Filter_Number*] | Setpoint_Function_Name = *Setpoint_Function_Name* | Control_Variable = *Control_Variable* | [Enable_Clamping = *enable_clamping*] | Max_Cooling_Power = *Max_Cooling_Power*)

Summary

This is similar to PID_Controlled, however, the flux is applied to cool the volume instead of heat it. Max_Cooling_Power should be a positive value. See description for PID_CONTROLLED for more details

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output

If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Flux_Output

If specified, the average flux of the given BC/Src is calculated and stored into the global variable name specified as the *flux_output* argument.

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
power_output	"string"	–
flux_output	"string"	–
toggle	"string"	–
p	real	0
i	real	0
d	real	0
t_i	real	0
t_d	real	0
Band	real	0
Filter_Tau	real	1
Filter_Number	real	1
Setpoint_Function_Name	"string"	–
Control_Variable	"string"	–
enable_clamping	integer	–
Max_Cooling_Power	real	–

Polynomial

Syntax

Bc No_Coverage_Flux For Energy ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Power_Output = *power_output*] | [Flux_Output = *flux_output*] | [Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The flux, F, is calculated as $F = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The

meaning of each option is as follows:

Power_Output
: If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Flux_Output
: If specified, the average flux of the given BC/Src is calculated and stored into the global variable name specified as the *flux_output* argument.

Toggle
: If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
power_output	“string”	–
flux_output	“string”	–
toggle	“string”	–
variable	“string”	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Rad

Syntax

Bc No_Coverage_Flux For Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Rad ([Power_Output = *power_output*] | [Flux_Output = *flux_output*] | [Toggle = *toggle*] | Crad = *crad* | T_Ref = *t_ref*)

Summary

Radiative energy flux defined with constant models for emissivity and reference temperature.

Description

The applied flux is given by:

$$c_{rad}\left(T^4 - T_{ref}^4\right)$$

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output

If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Flux_Output

If specified, the average flux of the given BC/Src is calculated and stored into the global variable name specified as the *flux_output* argument.

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
power_output	"string"	–
flux_output	"string"	–
toggle	"string"	–
crad	real	–
t_ref	real	–

Ray_Trace

Syntax

Bc No_Coverage_Flux For Energy ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Ray_Trace ([Power_Output = *power_output*] | [Flux_Output = *flux_output*] | [Toggle = *toggle*] | Ray_Source = *ray_source* | [Emission = *emission*])

Summary

Apply a ray tracing energy flux

Description

Adds an energy flux based on the intersection of rays defined in a ray tracing source block for the equation system with name given by the RAY_SOURCE parameter with the surface. The optional EMISSION parameter also imposes a σT^4 blackbody radiative emission term

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output

If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Flux_Output

If specified, the average flux of the given BC/Src is calculated and stored into the global variable name specified as the *flux_output* argument.

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
power_output	"string"	–
flux_output	"string"	–
toggle	"string"	–
ray_source	"string"	–
emission	"string"	FALSE

Rte_Field

Syntax

Bc No_Coverage_Flux For Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Rte_Field ([Power_Output = *power_output*] | [Flux_Output = *flux_output*] | [Toggle = *toggle*])

Summary

RTE energy flux from a field

Description

Flux is given by -f, where f is the field named 'rte_radiative_flux'

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output

If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Flux_Output

If specified, the average flux of the given BC/Src is calculated and

stored into the global variable name specified as the *flux_output* argument.

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
power_output	"string"	–
flux_output	"string"	–
toggle	"string"	–

Rte_Sp

Syntax

Bc No_Coverage_Flux For Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Rte_Sp ([Power_Output = *power_output*] | [Flux_Output = *flux_output*] | [Toggle = *toggle*] | Order = *order*)

Summary

RTE flux for energy equation with Simplified Spherical Harmonic RTE model with order ORDER

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output

If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Flux_Output

If specified, the average flux of the given BC/Src is calculated and stored into the global variable name specified as the *flux_output*

argument.

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
power_output	"string"	–
flux_output	"string"	–
toggle	"string"	–
order	integer	–

Scalar_String_Function

Syntax

Bc No_Coverage_Flux For Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Power_Output = *power_output*] | [Flux_Output = *flux_output*] | [Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output

If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Flux_Output

If specified, the average flux of the given BC/Src is calculated and stored into the global variable name specified as the *flux_output* argument.

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
power_output	"string"	–
flux_output	"string"	–
toggle	"string"	–
f	"string"	–

Scaled_Nat_Conv

Syntax

Bc No_Coverage_Flux For Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scaled_Nat_Conv ([Power_Output = *power_output*] | [Flux_Output = *flux_output*] | [Toggle = *toggle*] | [Multiplier = *multiplier*] | Scaling_Field = *scaling_field* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Scaled natural convection energy flux

Description

Flux is given by:

$$(M)(SC)(GV)(f)fcn(t)h\left(T-T_{ref}\right)$$

M is given by MULTIPLIER, SC by SCALING, GV by the value of the global variable with name GLOBAL_VAR, fcn is a function with name TIME_FUNCTION, and f is a field with name SCALING_FIELD. h and T_{ref} are taken from the HEAT TRANSFER COEFFICIENT and BC REFERENCE TEMPERATURE expressions.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output

If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Flux_Output

If specified, the average flux of the given BC/Src is calculated and stored into the global variable name specified as the *flux_output* argument.

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
power_output	“string”	–
flux_output	“string”	–
toggle	“string”	–
multiplier	real	1
scaling_field	“string”	–
scaling	real	1
global_var	“string”	–
time_function	“string”	–

Sharp_Power_Weld

Syntax

Bc No_Coverage_Flux For Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Sharp_Power_Weld ([Power_Output = *power_output*] | [Flux_Output = *flux_output*] | [Toggle = *toggle*] | R = *r* | [Clip_Radius = *clip_radius*] | [Efficiency = *efficiency*] | Origin_And_Normal_Function = *origin_and_normal_function* | [Old_Temperature = *old_temperature*] | Path_Function = *path_function* | [Deposition = *deposition*] | [Integrated = *integrated*] | [Clip = *clip*])

Summary

Energy flux due to an incident 'top hat' beam

Description

The coordinates of the center of the beam are provided by the X, Y, and Z components of the provided origin_and_normal_function. The direction of the beam is provided by the NX, NY, and NZ components of the origin_and_normal_function. The applied flux is given by:

$$\frac{(-\boldsymbol{n}) \cdot \boldsymbol{d}}{|\boldsymbol{d}|} \frac{\alpha e P}{\pi R^2}$$

P is the laser power defined for the material, R is the beam radius, α is the laser absorptivity defined for the material, e is the EFFICIENCY, and the term $-\boldsymbol{n} \cdot \boldsymbol{d}$ accounts for the fact that the surface may not be orthogonal to the laser. Points with values of r_p > CLIP_RADIUS will receive zero flux. r_p is the shortest distance between a point on the surface and the line defined by the beam origin and normal. If ACTIVATION is set to true, points whose temperature exceed ACTIVATION_TEMPERATURE will be flagged as activated for use with the ACTIVATION USER FUNCTION thermal conductivity model. If INTEGRATED is true, this flux will be integrated between the previous time step and the current one and the average flux applied.

See the *Directed Energy* section for the appropriate syntax to use this flux in the directed energy block.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output

If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Flux_Output

If specified, the average flux of the given BC/Src is calculated and stored into the global variable name specified as the *flux_output* argument.

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
power_output	“string”	–
flux_output	“string”	–
toggle	“string”	–
r	real	–
clip_radius	real	0
efficiency	real	1
origin_and_normal_function	“string”	–
old_temperature	“string”	TRUE
path_function	“string”	–
deposition	“string”	FALSE
integrated	“string”	FALSE
clip	“string”	TRUE

Sharp_Ray_Trace

Syntax

Bc No_Coverage_Flux For Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Sharp_Ray_Trace ([Power_Output = *power_output*] | [Flux_Output = *flux_output*] | [Toggle = *toggle*] | Path_Function = *path_function* | Dir = *dir* | Power = *power* | Type = *type* | [N_Rays = *n_rays*] | [N_R = *n_r*] | [N_Theta_Initial = *n_theta_initial*] | [N_Intersections = *n_intersections*] | [Weight_Ratio = *weight_ratio*] | [Id = *id*] | [Absorption_Model = *absorption_model*] | [Parallel_Type = *parallel_type*] | R = *r*)

Summary

Ray tracing implementation of a sharp circular beam energy flux

Description

r - Laser spot radius.

type - Either radial discretization or Monte Carlo. Determines how beam will be discretized into rays. Radial discretization is deterministic and breaks beams into approximately equal area segments radially. Each segment generates one

ray. The power of each ray varies based on which segment it is generated from. Monte Carlo is stochastic and will generate a user-specified number of rays distributed spatially according to the laser's shape function, with each ray having equal power of power/n_rays.

n_r - Number of segments in radial direction for radial discretization.

n_theta_initial - Number of segments in theta direction in the smallest radial segment. The number of theta segments will grow as radial direction increases so segments have approximately the same area. Total number of rays will be n_r*n_r*n_theta_initial.

n_rays - Number of rays to generate for Monte Carlo discretization.

path_function - Function giving x, y, z coordinates of laser center as function of time.

dir - Function giving x,y,z components of direction that rays will travel initially from laser center defined by path_function.

power - Function giving total laser power.

n_intersections - Maximum number of times to intersect a ray with mesh elements due to reflections (default: 10).

weight_ratio - Fraction of initial ray power below which rays will be considered spent and no longer be traced (default: 0.001).

absorption_model - Either Specular, to use a surface emissivity value to determine fraction of energy absorbed and reflected with each ray intersection or Fresnel to use Fresnel model dependent on ray angle of incidence.

parallel_type - Either ray_parallel, to divide rays between processors with a globally shared mesh part or scene_parallel, to divide mesh part between processors with a globally shared set of rays.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output
: If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Flux_Output
: If specified, the average flux of the given BC/Src is calculated and stored into the global variable name specified as the *flux_output* argument.

Toggle
: If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
power_output	“string”	–
flux_output	“string”	–
toggle	“string”	–
path_function	“string”	–
dir	“string”	–
power	“string”	–
type	“string”	–
n_rays	integer	0
n_r	integer	0
n_theta_initial	integer	0
n_intersections	integer	10
weight_ratio	real	0.001
id	“string”	–
absorption_model	“string”	SPECULAR
parallel_type	“string”	RAY_PARALLEL
r	real	–

Thermal_Latent_Heat

Syntax

Bc No_Coverage_Flux For Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Thermal_Latent_Heat ([Power_Output = *power_output*] | [Flux_Output = *flux_output*] | [Toggle = *toggle*])

Summary

Thermal latent heat energy flux

Description

Flux is given by:

$$H_v v_e$$

H_v is the expression for latent heat of vaporization and v_e is the local interface speed

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output
If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Flux_Output
If specified, the average flux of the given BC/Src is calculated and stored into the global variable name specified as the *flux_output* argument.

Toggle
If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
power_output	"string"	–
flux_output	"string"	–
toggle	"string"	–

User_Field

Syntax

Bc No_Coverage_Flux For Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Power_Output = *power_output*] | [Flux_Output = *flux_output*] | [Toggle = *toggle*] | [Multiplier = *multiplier*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

User field energy flux

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output

If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Flux_Output

If specified, the average flux of the given BC/Src is calculated and stored into the global variable name specified as the *flux_output* argument.

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
power_output	"string"	–
flux_output	"string"	–
toggle	"string"	–
multiplier	real	1
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc No_Coverage_Flux For Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Power_Output = *power_output*] | [Flux_Output = *flux_output*] | [Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] |

[Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output

If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Flux_Output

If specified, the average flux of the given BC/Src is calculated and stored into the global variable name specified as the *flux_output* argument.

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
power_output	“string”	–
flux_output	“string”	–
toggle	“string”	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc No_Coverage_Flux For Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Power_Output = *power_output*] | [Flux_Output = *flux_output*] | [Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output

If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Flux_Output

If specified, the average flux of the given BC/Src is calculated and stored into the global variable name specified as the *flux_output* argument.

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
power_output	“string”	–
flux_output	“string”	–
toggle	“string”	–
Name	“string”	–
plugin_parameters	“string”	–

User_Vector_Field

Syntax

Bc No_Coverage_Flux For Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Vector_Field ([Power_Output = *power_output*] | [Flux_Output = *flux_output*] | [Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*] | [Multiplier = *multiplier*])

Summary

User vector field energy flux

Description

Flux is given by:

$$(M)(SC)(GV)fcn(t)\boldsymbol{f}\cdot\boldsymbol{n}$$

M is given by MULTIPLIER, SC by SCALING, GV by the value of the global variable with name GLOBAL_VAR, fcn is a function with name TIME_FUNCTION, f is a field with name NAME, and n is the surface normal

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output

If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Flux_Output

If specified, the average flux of the given BC/Src is calculated and stored into the global variable name specified as the *flux_output* argument.

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
power_output	"string"	–
flux_output	"string"	–
toggle	"string"	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–
multiplier	real	1

User_Vector_Field_Influx

Syntax

Bc No_Coverage_Flux For Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Vector_Field_Influx ([Power_Output = *power_output*] | [Flux_Output = *flux_output*] | [Toggle = *toggle*] | [Multiplier = *multiplier*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

User vector field influx energy flux

Description

Flux is given by:

$$(M)(SC)(GV)fcn(t)\boldsymbol{f}\cdot\boldsymbol{n}$$

M is given by MULTIPLIER, SC by SCALING, GV by the value of the global variable with name GLOBAL_VAR, fcn is a function with name TIME_FUNCTION, f is a field with name NAME, and n is the surface normal. Flux will be capped at zero (inflow only)

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output

If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Flux_Output

If specified, the average flux of the given BC/Src is calculated and stored into the global variable name specified as the *flux_output* argument.

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
power_output	“string”	–
flux_output	“string”	–
toggle	“string”	–
multiplier	real	1
name	“string”	–
scaling	real	1
global_var	“string”	–
time_function	“string”	–

Vapor_Cooling

Syntax

Bc No_Coverage_Flux For Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Vapor_Cooling ([Power_Output = *power_output*] | [Flux_Output = *flux_output*] | [Toggle = *toggle*] | Tboil = *tboil*)

Summary

Empirical relation for cooling due to vaporization of stainless steel.

Description

Flux is given by:

$$814373(T - T_{boil}) - 2248.31(T - T_{boil})^2 + 27.1683(T - T_{boil})^3$$

For $T - T_{boil}$ less than 170 and:

$$-310360000 + 3.272400(T - T_{boil}) - 1808.4(T - T_{boil})^2 + 2.7284(T - T_{boil})^3$$

For $T - T_{boil}$ greater than 170. For $T - T_{boil}$ less than zero, no flux is applied

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output

If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Flux_Output

If specified, the average flux of the given BC/Src is calculated and stored into the global variable name specified as the *flux_output* argument.

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
power_output	"string"	–
flux_output	"string"	–
toggle	"string"	–
tboil	real	–

Enthalpy

Constant

Syntax

Bc No_Coverage_Flux For Enthalpy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | Value = *value*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
value	real	–

Convective_Outflow

Syntax

Bc No_Coverage_Flux For Enthalpy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Convective_Outflow [Toggle = *toggle*]

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–

Exponential

Syntax

Bc No_Coverage_Flux For Enthalpy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Generalized_Nat_Conv

Syntax

Bc No_Coverage_Flux For Enthalpy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Generalized_Nat_Conv ([Toggle = *toggle*] | [Multiplier = *multiplier*])

Summary

Generalized natural convection enthalpy flux

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
multiplier	real	1

Generalized_Rad

Syntax

Bc No_Coverage_Flux For Enthalpy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Generalized_Rad ([Toggle = *toggle*] | [Multiplier = *multiplier*] | [T_Ref = *t_ref*])

Summary

Radiative energy flux, with values for emissivity, form factor and irradiation taken from expressions

Description

Flux is:

$$M\epsilon f\left(\sigma T^4 - I\right)$$

M is MULTIPLIER. σ is the Stefan-Boltzmann constant. ϵ and F are expressions for emissivity and form factor. T is surface temperature. I is the expression for irradiation. If I does not already exist, it will be constructed as σT_{ref}^4, where T_{ref} is either provided by the BC Rad Reference Temperature expression or the inline T_{ref} parameter (constant or string function).

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
multiplier	real	1
t_ref	"string"	–

Global

Syntax

Bc No_Coverage_Flux For Enthalpy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
global_name	"string"	–
component	integer	0

Nat_Conv

Syntax

Bc No_Coverage_Flux For Enthalpy ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nat_Conv ([Toggle = *toggle*] | H = *h* | T_Ref = *t_ref*)

Summary

Natural convection BC for enthalpy flux

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
h	real	–
t_ref	real	–

Nd_Table

Syntax

Bc No_Coverage_Flux For Enthalpy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc No_Coverage_Flux For Enthalpy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Rad

Syntax

Bc No_Coverage_Flux For Enthalpy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Rad ([Toggle = *toggle*] | Crad = *crad* | T_Ref = *t_ref*)

Summary

Radiative energy flux defined with constant models for emissivity and reference temperature.

Description

The applied flux is given by:

$$c_{rad}\left(T^4 - T_{ref}^4\right)$$

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
crad	real	–
t_ref	real	–

Scalar_String_Function

Syntax

Bc No_Coverage_Flux For Enthalpy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
f	“string”	–

User_Field

Syntax

Bc No_Coverage_Flux For Enthalpy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc No_Coverage_Flux For Enthalpy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc No_Coverage_Flux For Enthalpy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

Level_Set

Constant

Syntax

Bc No_Coverage_Flux For Level_Set ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | *{flux | value}* = *flux*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
flux	real	–

Exponential

Syntax

Bc No_Coverage_Flux For Level_Set ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc No_Coverage_Flux For Level_Set ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
global_name	"string"	–
component	integer	0

Inflow_Outflow

Syntax

Bc No_Coverage_Flux For Level_Set ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Inflow_Outflow [Toggle = *toggle*]

Summary

This model has been deprecated and removed from the code.

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–

Nd_Table

Syntax

Bc No_Coverage_Flux For Level_Set ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc No_Coverage_Flux For Level_Set ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
variable	“string”	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc No_Coverage_Flux For Level_Set ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
f	“string”	–

User_Field

Syntax

Bc No_Coverage_Flux For Level_Set ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
scaling	real	1
global_var	“string”	–
time_function	“string”	–

User_Function

Syntax

Bc No_Coverage_Flux For Level_Set ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc No_Coverage_Flux For Level_Set ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

Lumped_Div_Projection

Constant

Syntax

Bc No_Coverage_Flux For Lumped_Div_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | [Valuex = *valuex*] | [Valuey = *valuey*] | [Valuez = *valuez*])

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
valuex	real	0
valuey	real	0
valuez	real	0

Vector_String_Function

Syntax

Bc No_Coverage_Flux For Lumped_Div_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Vector_String_Function ([Toggle = *toggle*] | F_X = *f_x* | F_Y = *f_y* | [F_Z = *f_z*])

Summary

Value from a string function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f_x	"string"	–
f_y	"string"	–
f_z	"string"	–

Lumped_Div_Projection_X

Constant

Syntax

Bc No_Coverage_Flux For Lumped_Div_Projection_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | Value = *value*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
value	real	–

Exponential

Syntax

Bc No_Coverage_Flux For Lumped_Div_Projection_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
variable	“string”	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc No_Coverage_Flux For Lumped_Div_Projection_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Bc No_Coverage_Flux For Lumped_Div_Projection_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc No_Coverage_Flux For Lumped_Div_Projection_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc No_Coverage_Flux For Lumped_Div_Projection_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f	"string"	–

User_Field

Syntax

Bc No_Coverage_Flux For Lumped_Div_Projection_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
scaling	real	1
global_var	“string”	–
time_function	“string”	–

User_Function

Syntax

Bc No_Coverage_Flux For Lumped_Div_Projection_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc No_Coverage_Flux For Lumped_Div_Projection_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

Lumped_Div_Projection_Y

Constant

Syntax

Bc No_Coverage_Flux For Lumped_Div_Projection_Y ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | Value = *value*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
value	real	–

Exponential

Syntax

Bc No_Coverage_Flux For Lumped_Div_Projection_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
variable	“string”	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc No_Coverage_Flux For Lumped_Div_Projection_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Bc No_Coverage_Flux For Lumped_Div_Projection_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc No_Coverage_Flux For Lumped_Div_Projection_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc No_Coverage_Flux For Lumped_Div_Projection_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f	"string"	–

User_Field

Syntax

Bc No_Coverage_Flux For Lumped_Div_Projection_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc No_Coverage_Flux For Lumped_Div_Projection_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc No_Coverage_Flux For Lumped_Div_Projection_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

Lumped_Div_Projection_Z

Constant

Syntax

Bc No_Coverage_Flux For Lumped_Div_Projection_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | Value = *value*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
value	real	–

Exponential

Syntax

Bc No_Coverage_Flux For Lumped_Div_Projection_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc No_Coverage_Flux For Lumped_Div_Projection_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Bc No_Coverage_Flux For Lumped_Div_Projection_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc No_Coverage_Flux For Lumped_Div_Projection_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc No_Coverage_Flux For Lumped_Div_Projection_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f	"string"	–

User_Field

Syntax

Bc No_Coverage_Flux For Lumped_Div_Projection_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc No_Coverage_Flux For Lumped_Div_Projection_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc No_Coverage_Flux For Lumped_Div_Projection_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

Mass_Balance

Co2_Convective_Outflow

Syntax

Bc No_Coverage_Flux For Mass_Balance ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Co2_Convective_Outflow ([Toggle = *toggle*] | [Ref_Frac = *ref_frac*])

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
ref_frac	real	0

Constant

Syntax

Bc No_Coverage_Flux For Mass_Balance ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | Value = *value*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
value	real	–

Constant_Velocity

Syntax

Bc No_Coverage_Flux For Mass_Balance ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant_Velocity ([Toggle = *toggle*] | Mult = *Mult*)

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is

active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Mult	real	–

Convective_Outflow

Syntax

Bc No_Coverage_Flux For Mass_Balance ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Convective_Outflow ([Toggle = *toggle*] | [Ref_Frac = *ref_frac*])

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
ref_frac	real	0

Exponential

Syntax

Bc No_Coverage_Flux For Mass_Balance ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
variable	“string”	–
constant	real	0
multiplier	real	1
exponent	real	–

Fluid_Robin_Coupled_One_Region

Syntax

Bc No_Coverage_Flux For Mass_Balance ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Fluid_Robin_Coupled_One_Region ([Toggle = *toggle*] | [Coeff_Scaling = *coeff_scaling*])

Summary

Boundary condition for coupling of continuity to a free fluid in the same region.

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
coeff_scaling	real	0.001

Global

Syntax

Bc No_Coverage_Flux For Mass_Balance ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
global_name	"string"	–
component	integer	0

Mass_Fluid_Robin_Coupled

Syntax

Bc No_Coverage_Flux For Mass_Balance ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Mass_Fluid_Robin_Coupled ([Toggle = *toggle*] | [Coeff_Scaling = *coeff_scaling*] | P_Field = *P_Field* | Massflux_Field = *MassFlux_Field*)

Summary

Boundary condition for loose coupling of continuity to a free fluid region

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
coeff_scaling	real	0.001
P_Field	"string"	–
MassFlux_Field	"string"	–

Mass_Fraction_Fluid_Robin_Coupled

Syntax

Bc No_Coverage_Flux For Mass_Balance ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Mass_Fraction_Fluid_Robin_Coupled ([Toggle = *toggle*] | Mass_Fraction_Field = *mass_fraction_field* | Mass_Fraction_Diffusive_Flux_Field = *mass_fraction_diffusive_flux_field* | Coeff_Field = *coeff_field*)

Summary

Boundary condition for loose coupling of scalars to a free fluid region

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
mass_fraction_field	"string"	–
mass_fraction_diffusive_flux_field	"string"	–
coeff_field	"string"	–

Mass_Fraction_Fluid_Robin_Coupled_One_Region

Syntax

Bc No_Coverage_Flux For Mass_Balance ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Mass_Fraction_Fluid_Robin_Coupled_One_Region [Toggle = *toggle*]

Summary

Boundary condition for coupling of scalars to a free fluid in the same region.

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–

Mass_Open

Syntax

Bc No_Coverage_Flux For Mass_Balance ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Mass_Open ([Toggle = *toggle*] | [Pressure = *pressure*])

Summary

A weakly enforced pressure condition for open flow from a porous flow domain

Description

This model weakly enforces a pressure condition on the boundary of a porous domain, either equal to the specified constant `PRESSURE`, or set using the `BOUNDARY_PRESSURE` model from the surface material.

Note: This should only be used with porous flow equations.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
pressure	real	0

Mesh_Motion

Syntax

Bc No_Coverage_Flux For Mass_Balance ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Mesh_Motion [Toggle = *toggle*]

Summary

Moving boundary condition for Arbitrary Lagrangian-Eulerian (ALE) mesh motion in relative flux form. Accounts for the boundary advective flux due to general mesh motion.

Description

Applies the following boundary flux

$$-\int q\hat{\mathbf{u}}\cdot\mathbf{n}dS,$$

where $\hat{u}$ is the mesh velocity and where the integrated scalar q depends on the equation. For the gas phase `porous_enthalpy` equation with a `transported enthalpy = porous`, $q = \phi\rho_g h_g$. Similarly, for the gas phase `mass_balance` equation, $q = \phi\rho_g$ is the bulk mass density.

This flux boundary condition is automatically applied to the mass balance and porous enthalpy equations when the `use relative mass flux` porous flow option is turned on.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–

Moving_Boundary

Syntax

Bc No_Coverage_Flux For Mass_Balance ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Moving_Boundary [Toggle = *toggle*]

Summary

No penetration moving boundary condition, valid for Lagrangian mesh motion in non-relative flux form or relative flux form.

Description

Applies the following boundary flux

$$\int q\hat{\mathbf{u}} \cdot \mathbf{n} dS,$$

where $\hat{u}$ is the mesh velocity and where the integrated scalar q depends on the equation. If we consider the gas phase `porous_enthalpy` equation with a `transported enthalpy = porous`, then $q = \phi\rho_g h_g$. Similarly, for the gas phase `mass_balance` equation, $q = \phi\rho_g$ is the bulk mass density.

If we consider the gas phase `mass_balance` equation, with a gas phase porous velocity relative to the moving solid phase, then physically this boundary condition is derived from applying a no flux condition at moving boundary i.e.,

$$\mathbf{u}_g \cdot \mathbf{n} = \phi\hat{\mathbf{u}} \cdot \mathbf{n}.$$

In the Lagrangian setting, this flux is required to impose a no flux condition at the moving boundary with both the non-relative form (default) and relative form of the advection term. When using the relative form, typically a do-nothing condition would be sufficient to impose no flux; however, to simplify general ALE mesh motion for porous flow equations (i.e., mass balance and porous enthalpy) we automatically provide the mesh motion flux on boundaries for blocks that have mesh motion. Consequently, to impose a zero flux, this term is needed for cancellation.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–

Nd_Table

Syntax

Bc No_Coverage_Flux For Mass_Balance ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Open

Syntax

Bc No_Coverage_Flux For Mass_Balance ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Open ([Toggle = *toggle*] | [Pressure = *pressure*] | [Entrained_Value = *entrained_value*])

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
pressure	real	0
entrained_value	real	0

Open_Flow

Syntax

Bc No_Coverage_Flux For Mass_Balance ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Open_Flow [Toggle = *toggle*]

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–

Polynomial

Syntax

Bc No_Coverage_Flux For Mass_Balance ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
variable	“string”	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc No_Coverage_Flux For Mass_Balance ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
f	“string”	–

User_Field

Syntax

Bc No_Coverage_Flux For Mass_Balance ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc No_Coverage_Flux For Mass_Balance ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc No_Coverage_Flux For Mass_Balance ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

Well_Outflow

Syntax

Bc No_Coverage_Flux For Mass_Balance ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Well_Outflow ([Toggle = *toggle*] | Pi = *pi* | Pref = *pref*)

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
pi	"string"	–
pref	"string"	–

Mesh

Constant

Syntax

Bc No_Coverage_Flux For Mesh ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | [Valuex = *valuex*] | [Valuey = *valuey*] | [Valuez = *valuez*])

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
valuex	real	0
valuey	real	0
valuez	real	0

Constant_Traction

Syntax

Bc No_Coverage_Flux For Mesh ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant_Traction ([Toggle = *toggle*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Constant traction mesh flux

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
x	real	0
y	real	0
z	real	0

Damping

Syntax

Bc No_Coverage_Flux For Mesh ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Damping ([Toggle = *toggle*] | [Scale_Factor = *scale_factor*])

Summary

Dampen surface displacements.

Description

$$F = \frac{-2\mu s}{h}\frac{\partial u}{\partial t}$$

where 2μ is the Lame constant, s the optional "scale_factor" parameter, h the mesh size, and u the mesh displacements. This BC is primarily intended for use in conjunction with a rotated BC that enforces the normal mesh displacement, and serves to dampen tangential mesh displacements on the surface.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
scale_factor	real	1

Electric_Traction

Syntax

Bc No_Coverage_Flux For Mesh ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Electric_Traction ([Toggle = *toggle*] | [Sign = *sign*])

Summary

Electric traction mesh flux

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
sign	real	1

Penalized_Normal_Displacement

Syntax

Bc No_Coverage_Flux For Mesh ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Penalized_Normal_Displacement ([Toggle = *toggle*] | [Stiffness_Factor = *stiffness_factor*])

Summary

Penalized normal flux

Description

Enforces a weak zero-normal-displacement condition by applying a mesh flux of

$$\vec{F} = -2\mu S_f \left(\vec{D} \cdot \vec{n}\right) \vec{n}$$

Where 2μ is the mesh `TWO_MU` value, and S_f is the stiffness factor.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
stiffness_factor	real	1e+06

Pressure

Syntax

Bc No_Coverage_Flux For Mesh ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Pressure ([Toggle = *toggle*] | [User_Field_Name = *user_field_name*] | [Use_Bulk_Node = *use_bulk_node*] | [P = *p*] | [C_T = *c_t*] | [C_X = *c_x*] | [C_Y = *c_y*] | [C_Z = *c_z*] | [C_Cos_W = *c_cos_w*] | [C_Sin_W = *c_sin_w*] | [C1_Cos = *c1_cos*] | [C1_Sin = *c1_sin*])

Summary

Pressure mesh flux

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
user_field_name	"string"	–
use_bulk_node	"string"	–
p	real	0
c_t	real	0
c_x	real	0
c_y	real	0
c_z	real	0
c_cos_w	real	0
c_sin_w	real	0
c1_cos	real	0
c1_sin	real	0

Pressure_User_Function

Syntax

Bc No_Coverage_Flux For Mesh ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Pressure_User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Pressure user function mesh flux

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1

Shear_Free

Syntax

Bc No_Coverage_Flux For Mesh ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Shear_Free [Toggle = *toggle*]

Summary

Shear-free traction mesh flux

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–

Specified_Normal_Velocity

Syntax

Bc No_Coverage_Flux For Mesh ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Specified_Normal_Velocity ([Toggle = *toggle*] | [Stiffness_Factor = *stiffness_factor*])

Summary

Penalized normal mesh velocity

Description

Enforces a weak specified normal velocity condition by applying a mesh flux of

$$\vec{F} = 2\mu S_f \left(V_m - \dot{\vec{D}} \cdot \vec{n} \right) \vec{n}$$

Where 2μ is the mesh `TWO_MU` value, and S_f is the stiffness factor. The normal mesh velocity, V_m is provided by the `NORMAL_MESH_VELOCITY` expression on the surface (scalar, positive for outward mesh velocity).

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
stiffness_factor	real	1e+06

Transient_Traction

Syntax

Bc No_Coverage_Flux For Mesh ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Transient_Traction ([Toggle = *toggle*] | [A_X = *a_x*] | [A_Y = *a_y*] | [A_Z = *a_z*] | [B_X = *b_x*] | [B_Y = *b_y*] | [B_Z = *b_z*])

Summary

Transient traction mesh flux

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
a_x	real	0
a_y	real	0
a_z	real	0
b_x	real	0
b_y	real	0
b_z	real	0

Vector_String_Function

Syntax

Bc No_Coverage_Flux For Mesh ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Vector_String_Function ([Toggle = *toggle*] | F_X = *f_x* | F_Y = *f_y* | [F_Z = *f_z*])

Summary

Value from a string function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
f_x	“string”	–
f_y	“string”	–
f_z	“string”	–

Mesh_X

Constant

Syntax

Bc No_Coverage_Flux For Mesh_X ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | Value = *value*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
value	real	–

Exponential

Syntax

Bc No_Coverage_Flux For Mesh_X ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc No_Coverage_Flux For Mesh_X ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Bc No_Coverage_Flux For Mesh_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc No_Coverage_Flux For Mesh_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc No_Coverage_Flux For Mesh_X ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f	"string"	–

User_Field

Syntax

Bc No_Coverage_Flux For Mesh_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc No_Coverage_Flux For Mesh_X ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc No_Coverage_Flux For Mesh_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

Mesh_Y

Constant

Syntax

Bc No_Coverage_Flux For Mesh_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | Value = *value*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
value	real	–

Exponential

Syntax

Bc No_Coverage_Flux For Mesh_Y ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc No_Coverage_Flux For Mesh_Y ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Bc No_Coverage_Flux For Mesh_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc No_Coverage_Flux For Mesh_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc No_Coverage_Flux For Mesh_Y ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f	"string"	–

User_Field

Syntax

Bc No_Coverage_Flux For Mesh_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc No_Coverage_Flux For Mesh_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc No_Coverage_Flux For Mesh_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

Mesh_Z

Constant

Syntax

Bc No_Coverage_Flux For Mesh_Z ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | Value = *value*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
value	real	–

Exponential

Syntax

Bc No_Coverage_Flux For Mesh_Z ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc No_Coverage_Flux For Mesh_Z ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Bc No_Coverage_Flux For Mesh_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc No_Coverage_Flux For Mesh_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc No_Coverage_Flux For Mesh_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f	"string"	–

User_Field

Syntax

Bc No_Coverage_Flux For Mesh_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc No_Coverage_Flux For Mesh_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc No_Coverage_Flux For Mesh_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

Mixture_Fraction

Constant

Syntax

Bc No_Coverage_Flux For Mixture_Fraction ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | *{flux | value}* = *flux*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
flux	real	–

Exponential

Syntax

Bc No_Coverage_Flux For Mixture_Fraction ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc No_Coverage_Flux For Mixture_Fraction ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Bc No_Coverage_Flux For Mixture_Fraction ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Open_Flow_Convection

Syntax

Bc No_Coverage_Flux For Mixture_Fraction ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Open_Flow_Convection ([Toggle = *toggle*] | Reference_Value = *Reference_Value*)

Summary

Open flow convection BC for the mixture fraction equation

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Reference_Value	real	–

Polynomial

Syntax

Bc No_Coverage_Flux For Mixture_Fraction ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc No_Coverage_Flux For Mixture_Fraction ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f	"string"	–

User_Field

Syntax

Bc No_Coverage_Flux For Mixture_Fraction ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc No_Coverage_Flux For Mixture_Fraction ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc No_Coverage_Flux For Mixture_Fraction ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

Momentum

Capillary

Syntax

Bc No_Coverage_Flux For Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Capillary [Toggle = *toggle*]

Summary

LS capillary momentum flux

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–

Capillary_Gap

Syntax

Bc No_Coverage_Flux For Momentum ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Capillary_Gap ([Toggle = *toggle*] | V_W = *v_w* | G = *g* | Theta = *theta*)

Summary

This model has been deprecated

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
v_w	real	–
g	real	–
theta	real	–

Constant

Syntax

Bc No_Coverage_Flux For Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
x	real	0
y	real	0
z	real	0

Constant_Traction

Syntax

Bc No_Coverage_Flux For Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant_Traction ([Toggle = *toggle*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
x	real	0
y	real	0
z	real	0

Darcy_Slip

Syntax

Bc No_Coverage_Flux For Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Darcy_Slip ([Toggle = *toggle*] | [Vs_X = *vs_x*] | [Vs_Y = *vs_y*] | [Vs_Z = *vs_z*] | K = *k*)

Summary

Darcy slip momentum flux

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
vs_x	real	0
vs_y	real	0
vs_z	real	0
k	real	–

Electric_Traction

Syntax

Bc No_Coverage_Flux For Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Electric_Traction ([Toggle = *toggle*] | [Sign = *sign*])

Summary

Electric traction momentum flux

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
sign	real	1

Equilibrium_Capillary_Gap

Syntax

Bc No_Coverage_Flux For Momentum ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Equilibrium_Capillary_Gap ([Toggle = *toggle*] | Theta = *theta*)

Summary

This model has been deprecated

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
theta	real	–

Exponential_Vapor_Recoil_Pressure

Syntax

Bc No_Coverage_Flux For Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential_Vapor_Recoil_Pressure ([Toggle = *toggle*] | Tboil = *tboil* | L = *l* | Pa = *pa* | [Beta = *beta*] | Mw = *mw* | [Tl = *tl*] | [Th = *th*])

Summary

Exponential vapor recoil pressure momentum flux

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
tboil	real	–
l	real	–
pa	real	–
beta	real	1
mw	real	–
tl	real	-1
th	real	-1

Fixed_Curvature

Syntax

Bc No_Coverage_Flux For Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Fixed_Curvature ([Toggle = *toggle*] | Curvature = *curvature*)

Summary

This model has been deprecated and removed from the code.

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
curvature	real	–

Flow_Hydrostatic

Syntax

Bc No_Coverage_Flux For Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Flow_Hydrostatic ([Toggle = *toggle*] | [P_Ref = *p_ref*] | [Gx = *gx*] | [Gy = *gy*] | [Gz = *gz*])

Summary

Flow hydrostatic momentum flux

Description

The following options are present in most boundary conditions/sources. The

meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
p_ref	real	0
gx	real	0
gy	real	0
gz	real	0

Free_Open_Flow

Syntax

Bc No_Coverage_Flux For Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Free_Open_Flow ([Toggle = *toggle*] | Pressure = *pressure*)

Summary

Open flow for momentum; apply the full stress

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
pressure	real	–

Lens_Particles_Pressure

Syntax

Bc No_Coverage_Flux For Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Lens_Particles_Pressure ([Toggle = *toggle*] | Src_X = *src_x* | Dir_X = *dir_x* | Vel_X = *vel_x* | Src_Y = *src_y* | Dir_Y = *dir_y* | Vel_Y = *vel_y* | [Src_Z = *src_z*] | [Dir_Z = *dir_z*] | [Vel_Z = *vel_z*] | Massflowrate = *massFlowRate* | T_Liq = *T_liq* | T_Solid = *T_solid* | [T_On = *t_on*] | [T_Off = *t_off*] | Particlespeed = *particleSpeed* | Radius = *radius*)

Summary

LENS particles pressure momentum flux

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
src_x	real	–
dir_x	real	–
vel_x	real	–
src_y	real	–
dir_y	real	–
vel_y	real	–
src_z	real	0
dir_z	real	0
vel_z	real	0
massFlowRate	real	–
T_liq	real	–
T_solid	real	–
t_on	real	0
t_off	real	1.798e+308
particleSpeed	real	–
radius	real	–

Ls_Capillary

Syntax

Bc No_Coverage_Flux For Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Ls_Capillary [Toggle = *toggle*]

Summary

LS capillary momentum flux

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–

Ls_Oriented_Slip

Syntax

Bc No_Coverage_Flux For Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Ls_Oriented_Slip ([Toggle = *toggle*] | [Vs_X = *vs_x*] | [Vs_Y = *vs_y*] | [Vs_Z = *vs_z*] | [Beta_Normal_A = *beta_normal_a*] | [Beta_Tangent_A = *beta_tangent_a*] | [Beta_Normal_B = *beta_normal_b*] | [Beta_Tangent_B = *beta_tangent_b*])

Summary

Oriented slip momentum flux

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
vs_x	real	0
vs_y	real	0
vs_z	real	0
beta_normal_a	real	0
beta_tangent_a	real	0
beta_normal_b	real	0
beta_tangent_b	real	0

Node_Normal_Capillary

Syntax

Bc No_Coverage_Flux For Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Node_Normal_Capillary [Toggle = *toggle*]

Summary

This model has been deprecated and removed from the code.

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–

Non_Ibp_Pressure

Syntax

Bc No_Coverage_Flux For Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Non_Ibp_Pressure ([Toggle = *toggle*] | [Pressure = *pressure*])

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
pressure	"string"	–

Open_Flow

Syntax

Bc No_Coverage_Flux For Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Open_Flow ([Toggle = *toggle*] | [Pressure = *pressure*] | [Total_Pressure = *total_pressure*])

Summary

Open flow for momentum; enforce normal gradient to be zero

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
pressure	real	0
total_pressure	real	0

Oriented_Slip

Syntax

Bc No_Coverage_Flux For Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Oriented_Slip ([Toggle = *toggle*] | [Vs_X = *vs_x*] | [Vs_Y = *vs_y*] | [Vs_Z = *vs_z*] | [Vs_Expr = *vs_expr*] | Beta_Normal = *beta_normal* | Beta_Tangent = *beta_tangent*)

Summary

Oriented slip momentum flux. Essentially a sum of the Navier-Slip condition

and a penalty condition for enforcing velocity in the normal direction.

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
vs_x	real	0
vs_y	real	0
vs_z	real	0
vs_expr	"string"	–
beta_normal	real	–
beta_tangent	real	–

Pressure

Syntax

Bc No_Coverage_Flux For Momentum ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Pressure ([Toggle = *toggle*] | [User_Field_Name = *user_field_name*] | [Use_Bulk_Node = *use_bulk_node*] | [P = *p*] | [C_T = *c_t*] | [C_X = *c_x*] | [C_Y = *c_y*] | [C_Z = *c_z*] | [C_Cos_W = *c_cos_w*] | [C_Sin_W = *c_sin_w*] | [C1_Cos = *c1_cos*] | [C1_Sin = *c1_sin*])

Summary

Pressure momentum flux

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
user_field_name	"string"	–
use_bulk_node	"string"	–
p	real	0
c_t	real	0
c_x	real	0
c_y	real	0
c_z	real	0
c_cos_w	real	0
c_sin_w	real	0
c1_cos	real	0
c1_sin	real	0

Pressure_User_Function

Syntax

Bc No_Coverage_Flux For Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Pressure_User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Pressure user function momentum flux

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is

active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1

Projected_Capillary

Syntax

Bc No_Coverage_Flux For Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Projected_Capillary [Toggle = *toggle*]

Summary

Projected capillary momentum flux

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–

Reconstructed_Curvature

Syntax

Bc No_Coverage_Flux For Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Reconstructed_Curvature ([Toggle = *toggle*] | [Ls_Size = *ls_size*] | [Fixed_Curvature = *fixed_curvature*])

Summary

This model has been deprecated and removed from the code.

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
ls_size	integer	0
fixed_curvature	real	-1

Simple_Spring_Force

Syntax

Bc No_Coverage_Flux For Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Simple_Spring_Force ([Toggle = *toggle*] | [Kx = *kx*] | [Ky = *ky*] | [Kz = *kz*] | [X0 = *x0*] | [Y0 = *y0*] | [Z0 = *z0*])

Summary

This model has been deprecated and removed from the code.

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
kx	real	0
ky	real	0
kz	real	0
x0	real	0
y0	real	0
z0	real	0

Slip

Syntax

Bc No_Coverage_Flux For Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Slip ([Toggle = *toggle*] | [Vs_X = *vs_x*] | [Vs_Y = *vs_y*] | [Vs_Z = *vs_z*] | Beta = *beta*)

Summary

Slip momentum flux

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
vs_x	real	0
vs_y	real	0
vs_z	real	0
beta	real	–

Slip_Length

Syntax

Bc No_Coverage_Flux For Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Slip_Length ([Toggle = *toggle*] | [Vs_X = *vs_x*] | [Vs_Y = *vs_y*] | [Vs_Z = *vs_z*] | Beta = *beta*)

Summary

Slip length momentum flux

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
vs_x	real	0
vs_y	real	0
vs_z	real	0
beta	real	–

Spring_Force

Syntax

Bc No_Coverage_Flux For Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Spring_Force ([Toggle = *toggle*] | Mult = *Mult*)

Summary

Spring force for momentum boundary condition

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Mult	real	–

Symmetry_Flow

Syntax

Bc No_Coverage_Flux For Momentum ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Symmetry_Flow [Toggle = *toggle*]

Summary

Symmetry flow momentum flux

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–

Transient_Traction

Syntax

Bc No_Coverage_Flux For Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Transient_Traction ([Toggle = *toggle*] | [A_X = *a_x*] | [A_Y = *a_y*] | [A_Z = *a_z*] | [B_X = *b_x*] | [B_Y = *b_y*] | [B_Z = *b_z*])

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
a_x	real	0
a_y	real	0
a_z	real	0
b_x	real	0
b_y	real	0
b_z	real	0

Vapor_Recoil_Pressure

Syntax

Bc No_Coverage_Flux For Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Vapor_Recoil_Pressure ([Toggle = *toggle*] | Tboil = *tboil*)

Summary

Vapor recoil pressure momentum flux

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
tboil	real	–

Vector_String_Function

Syntax

Bc No_Coverage_Flux For Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Vector_String_Function ([Toggle = *toggle*] | F_X = *f_x* | F_Y = *f_y* | [F_Z = *f_z*])

Summary

Value from a string function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
f_x	“string”	–
f_y	“string”	–
f_z	“string”	–

Momentum_X

Constant

Syntax

Bc No_Coverage_Flux For Momentum_X ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | Value = *value*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
value	real	–

Exponential

Syntax

Bc No_Coverage_Flux For Momentum_X ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc No_Coverage_Flux For Momentum_X ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Bc No_Coverage_Flux For Momentum_X ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc No_Coverage_Flux For Momentum_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc No_Coverage_Flux For Momentum_X ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f	"string"	–

User_Field

Syntax

Bc No_Coverage_Flux For Momentum_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc No_Coverage_Flux For Momentum_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc No_Coverage_Flux For Momentum_X ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

Momentum_Y

Constant

Syntax

Bc No_Coverage_Flux For Momentum_Y ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | Value = *value*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
value	real	–

Exponential

Syntax

Bc No_Coverage_Flux For Momentum_Y ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc No_Coverage_Flux For Momentum_Y ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Bc No_Coverage_Flux For Momentum_Y ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc No_Coverage_Flux For Momentum_Y ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc No_Coverage_Flux For Momentum_Y ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f	"string"	–

User_Field

Syntax

Bc No_Coverage_Flux For Momentum_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc No_Coverage_Flux For Momentum_Y ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc No_Coverage_Flux For Momentum_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

Momentum_Z

Constant

Syntax

Bc No_Coverage_Flux For Momentum_Z ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | Value = *value*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
value	real	–

Exponential

Syntax

Bc No_Coverage_Flux For Momentum_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc No_Coverage_Flux For Momentum_Z ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Bc No_Coverage_Flux For Momentum_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc No_Coverage_Flux For Momentum_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc No_Coverage_Flux For Momentum_Z ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f	"string"	–

User_Field

Syntax

Bc No_Coverage_Flux For Momentum_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc No_Coverage_Flux For Momentum_Z ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc No_Coverage_Flux For Momentum_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

Porous_Enthalpy

Constant

Syntax

Bc No_Coverage_Flux For Porous_Enthalpy ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | Value = *value*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
value	real	–

Convective_Outflow

Syntax

Bc No_Coverage_Flux For Porous_Enthalpy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Convective_Outflow ([Toggle = *toggle*] | [Ref_H = *ref_h*])

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
ref_h	real	0

Enclosure_Radiation

Syntax

Bc No_Coverage_Flux For Porous_Enthalpy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Enclosure_Radiation ([Toggle = *toggle*] | [Multiplier = *multiplier*] | Enclosure = *enclosure*)

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
multiplier	real	1
enclosure	“string”	–

Exponential

Syntax

Bc No_Coverage_Flux For Porous_Enthalpy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the

constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Fluid_Robin_Coupled

Syntax

Bc No_Coverage_Flux For Porous_Enthalpy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Fluid_Robin_Coupled ([Toggle = *toggle*] | Enthalpy_Field = *enthalpy_field* | Enthalpy_Diffusive_Flux_Field = *enthalpy_diffusive_flux_field* | Coeff_Field = *coeff_field*)

Summary

Boundary condition for loose coupling of enthalpy to a free fluid region

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
enthalpy_field	"string"	–
enthalpy_diffusive_flux_field	"string"	–
coeff_field	"string"	–

Fluid_Robin_Coupled_One_Region

Syntax

Bc No_Coverage_Flux For Porous_Enthalpy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Fluid_Robin_Coupled_One_Region [Toggle = *toggle*]

Summary

Boundary condition for loose coupling of enthalpy to a free fluid region

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–

Fluid_Robin_Coupled_With_Solid_Convection

Syntax

Bc No_Coverage_Flux For Porous_Enthalpy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Fluid_Robin_Coupled_With_Solid_Convection ([Toggle = *toggle*] | Enthalpy_Field = *enthalpy_field* | Enthalpy_Diffusive_Flux_Field = *enthalpy_diffusive_flux_field* | Coeff_Field = *coeff_field*)

Summary

Boundary condition for loose coupling of enthalpy to a free fluid region

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
enthalpy_field	“string”	–
enthalpy_diffusive_flux_field	“string”	–
coeff_field	“string”	–

Fluid_Robin_Coupled_With_Solid_Convection_One_Region

Syntax

Bc No_Coverage_Flux For Porous_Enthalpy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Fluid_Robin_Coupled_With_Solid_Convection_One_Region [Toggle = *toggle*]

Summary

Boundary condition for loose coupling of enthalpy to a free fluid region

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–

Fluid_Solid_Convection_Coupled

Syntax

Bc No_Coverage_Flux For Porous_Enthalpy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Fluid_Solid_Convection_Coupled ([Toggle = *toggle*] | [Multiplier = *multiplier*] | Fluid_Temperature_Field = *fluid_temperature_field*)

Summary

Boundary condition for loose coupling of enthalpy to a free fluid region

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
multiplier	real	1
fluid_temperature_field	"string"	–

Fluid_Solid_Convection_Coupled_One_Region

Syntax

Bc No_Coverage_Flux For Porous_Enthalpy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Fluid_Solid_Convection_Coupled_One_Region ([Toggle = *toggle*] | [Multiplier = *multiplier*])

Summary

Boundary condition for loose coupling of enthalpy to a free fluid region

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
multiplier	real	1

Generalized_Nat_Conv

Syntax

Bc No_Coverage_Flux For Porous_Enthalpy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Generalized_Nat_Conv ([Toggle = *toggle*] | [Multiplier = *multiplier*])

Summary

Generalized natural convection flux BC

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
multiplier	real	1

Generalized_Rad

Syntax

Bc No_Coverage_Flux For Porous_Enthalpy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Generalized_Rad ([Toggle = *toggle*] | [Multiplier = *multiplier*] | [T_Ref = *t_ref*])

Summary

Radiative energy flux, with values for emissivity, form factor and irradiation taken from expressions

Description

Flux is:

$$M\epsilon f\left(\sigma T^4 - I\right)$$

M is MULTIPLIER. σ is the Stefan-Boltzmann constant. ϵ and F are expressions for emissivity and form factor. T is surface temperature. I is the expression for irradiation. If I does not already exist, it will be constructed as σT_{ref}^4, where T_{ref} is either provided by the BC Rad Reference Temperature expression or the inline T_{ref} parameter (constant or string function).

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
multiplier	real	1
t_ref	“string”	–

Global

Syntax

Bc No_Coverage_Flux For Porous_Enthalpy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
global_name	“string”	–
component	integer	0

Material_Phase_Sum_Open

Syntax

Bc No_Coverage_Flux For Porous_Enthalpy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Material_Phase_Sum_Open ([Toggle = *toggle*] | [Pressure = *pressure*] | [Entrained_Value = *entrained_value*])

Summary

Open boundary condition for a single temperature model, wherein the phases are assumed to be in thermal equilibrium. The advective flux of enthalpy for

each phase is summed at the open.Must provide the BOUNDARY_ENTRAINED_ENTHALPY expression for entrainment.

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
pressure	real	0
entrained_value	real	0

Mesh_Motion

Syntax

Bc No_Coverage_Flux For Porous_Enthalpy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Mesh_Motion [Toggle = *toggle*]

Summary

Moving boundary condition for Arbitrary Lagrangian-Eulerian (ALE) mesh motion in relative flux form. Accounts for the boundary advective flux due to general mesh motion.

Description

Applies the following boundary flux

$$-\int q\hat{\mathbf{u}}\cdot\mathbf{n}dS,$$

where $\hat{u}$ is the mesh velocity and where the integrated scalar q depends on the equation. For the gas phase `porous_enthalpy` equation with a `transported`

`enthalpy = porous`, $q = \phi\rho_g h_g$. Similarly, for the gas phase `mass_balance` equation, $q = \phi\rho_g$ is the bulk mass density.

This flux boundary condition is automatically applied to the mass balance and porous enthalpy equations when the `use relative mass flux` porous flow option is turned on.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle
If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–

Moving_Boundary

Syntax

Bc No_Coverage_Flux For Porous_Enthalpy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Moving_Boundary [Toggle = *toggle*]

Summary

No penetration moving boundary condition, valid for Lagrangian mesh motion in non-relative flux form or relative flux form.

Description

Applies the following boundary flux

$$\int q\hat{\mathbf{u}} \cdot \mathbf{n} dS,$$

where $\hat{u}$ is the mesh velocity and where the integrated scalar q depends on the equation. If we consider the gas phase `porous_enthalpy` equation with a `transported enthalpy = porous`, then $q = \phi\rho_g h_g$. Similarly, for the gas phase `mass_balance` equation, $q = \phi\rho_g$ is the bulk mass density.

If we consider the gas phase `mass_balance` equation, with a gas phase porous velocity relative to the moving solid phase, then physically this boundary condition is derived from applying a no flux condition at moving boundary i.e.,

$$\mathbf{u}_g \cdot \mathbf{n} = \phi \hat{\mathbf{u}} \cdot \mathbf{n}.$$

In the Lagrangian setting, this flux is required to impose a no flux condition at the moving boundary with both the non-relative form (default) and relative form of the advection term. When using the relative form, typically a do-nothing condition would be sufficient to impose no flux; however, to simplify general ALE mesh motion for porous flow equations (i.e., mass balance and porous enthalpy) we automatically provide the mesh motion flux on boundaries for blocks that have mesh motion. Consequently, to impose a zero flux, this term is needed for cancellation.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–

Nd_Table

Syntax

Bc No_Coverage_Flux For Porous_Enthalpy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

One_Temp_Multiphase_Open

Syntax

Bc No_Coverage_Flux For Porous_Enthalpy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = One_Temp_Multiphase_Open ([Toggle = *toggle*] | [Pressure = *pressure*] | [Entrained_Value = *entrained_value*])

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
pressure	real	0
entrained_value	real	0

Open

Syntax

Bc No_Coverage_Flux For Porous_Enthalpy ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Open ([Toggle = *toggle*] | [Pressure = *pressure*] | [Entrained_Value = *entrained_value*])

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
pressure	real	0
entrained_value	real	0

Polynomial

Syntax

Bc No_Coverage_Flux For Porous_Enthalpy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
variable	“string”	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc No_Coverage_Flux For Porous_Enthalpy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
f	“string”	–

User_Field

Syntax

Bc No_Coverage_Flux For Porous_Enthalpy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc No_Coverage_Flux For Porous_Enthalpy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc No_Coverage_Flux For Porous_Enthalpy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

User_Vector_Field

Syntax

Bc No_Coverage_Flux For Porous_Enthalpy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Vector_Field ([Toggle = *toggle*] | Name = *name* | [Multiplier = *multiplier*])

Summary

User vector field enthalpy flux

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
multiplier	real	1

Porous_Species

Constant

Syntax

Bc No_Coverage_Flux For Porous_Species ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | *{flux | value}* = *flux*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
flux	real	–

Exponential

Syntax

Bc No_Coverage_Flux For Porous_Species ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc No_Coverage_Flux For Porous_Species ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
global_name	"string"	–
component	integer	0

Moving_Boundary

Syntax

Bc No_Coverage_Flux For Porous_Species ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Moving_Boundary [Toggle = *toggle*]

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–

Nd_Table

Syntax

Bc No_Coverage_Flux For Porous_Species ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Open_Flow

Syntax

Bc No_Coverage_Flux For Porous_Species ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Open_Flow [Toggle = *toggle*]

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–

Polynomial

Syntax

Bc No_Coverage_Flux For Porous_Species ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc No_Coverage_Flux For Porous_Species ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name.

Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
f	“string”	–

User_Field

Syntax

Bc No_Coverage_Flux For Porous_Species ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc No_Coverage_Flux For Porous_Species ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc No_Coverage_Flux For Porous_Species ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

Solid

Constant

Syntax

Bc No_Coverage_Flux For Solid ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | [Valuex = *valuex*] | [Valuey = *valuey*] | [Valuez = *valuez*])

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
valuex	real	0
valuey	real	0
valuez	real	0

Constant_Traction

Syntax

Bc No_Coverage_Flux For Solid ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant_Traction ([Toggle = *toggle*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Constant traction solid flux

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
x	real	0
y	real	0
z	real	0

Electric_Traction

Syntax

Bc No_Coverage_Flux For Solid ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Electric_Traction ([Toggle = *toggle*] | [Sign = *sign*])

Summary

Electric traction solid flux

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
sign	real	1

Pressure

Syntax

Bc No_Coverage_Flux For Solid ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Pressure ([Toggle = *toggle*] | [User_Field_Name = *user_field_name*] | [Use_Bulk_Node = *use_bulk_node*] | [P = *p*] | [C_T = *c_t*] | [C_X = *c_x*] | [C_Y = *c_y*] | [C_Z = *c_z*] | [C_Cos_W = *c_cos_w*] | [C_Sin_W = *c_sin_w*] | [C1_Cos = *c1_cos*] | [C1_Sin = *c1_sin*])

Summary

Pressure solid flux

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
user_field_name	"string"	–
use_bulk_node	"string"	–
p	real	0
c_t	real	0
c_x	real	0
c_y	real	0
c_z	real	0
c_cos_w	real	0
c_sin_w	real	0
c1_cos	real	0
c1_sin	real	0

Pressure_User_Function

Syntax

Bc No_Coverage_Flux For Solid ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Pressure_User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Pressure user function solid flux

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1

Shear_Free

Syntax

Bc No_Coverage_Flux For Solid ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Shear_Free [Toggle = *toggle*]

Summary

Shear-free traction solid flux

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–

Spring_Force

Syntax

Bc No_Coverage_Flux For Solid ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Spring_Force ([Toggle = *toggle*] | Mult = *Mult*)

Summary

Spring force solid flux

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Mult	real	–

Transient_Traction

Syntax

Bc No_Coverage_Flux For Solid ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Transient_Traction ([Toggle = *toggle*] | [A_X = *a_x*] | [A_Y = *a_y*] | [A_Z = *a_z*] | [B_X = *b_x*] | [B_Y = *b_y*] | [B_Z = *b_z*])

Summary

Transient traction solid flux

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
a_x	real	0
a_y	real	0
a_z	real	0
b_x	real	0
b_y	real	0
b_z	real	0

Vector_String_Function

Syntax

Bc No_Coverage_Flux For Solid ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Vector_String_Function ([Toggle = *toggle*] | F_X = *f_x* | F_Y = *f_y* | [F_Z = *f_z*])

Summary

Value from a string function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
f_x	“string”	–
f_y	“string”	–
f_z	“string”	–

Solid_X

Constant

Syntax

Bc No_Coverage_Flux For Solid_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | Value = *value*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
value	real	–

Exponential

Syntax

Bc No_Coverage_Flux For Solid_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc No_Coverage_Flux For Solid_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Bc No_Coverage_Flux For Solid_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc No_Coverage_Flux For Solid_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc No_Coverage_Flux For Solid_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f	"string"	–

User_Field

Syntax

Bc No_Coverage_Flux For Solid_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc No_Coverage_Flux For Solid_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc No_Coverage_Flux For Solid_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

Solid_Y

Constant

Syntax

Bc No_Coverage_Flux For Solid_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | Value = *value*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
value	real	–

Exponential

Syntax

Bc No_Coverage_Flux For Solid_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc No_Coverage_Flux For Solid_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Bc No_Coverage_Flux For Solid_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc No_Coverage_Flux For Solid_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc No_Coverage_Flux For Solid_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f	"string"	–

User_Field

Syntax

Bc No_Coverage_Flux For Solid_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc No_Coverage_Flux For Solid_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc No_Coverage_Flux For Solid_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

Solid_Z

Constant

Syntax

Bc No_Coverage_Flux For Solid_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | Value = *value*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
value	real	–

Exponential

Syntax

Bc No_Coverage_Flux For Solid_Z ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc No_Coverage_Flux For Solid_Z ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Bc No_Coverage_Flux For Solid_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc No_Coverage_Flux For Solid_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc No_Coverage_Flux For Solid_Z ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f	"string"	–

User_Field

Syntax

Bc No_Coverage_Flux For Solid_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc No_Coverage_Flux For Solid_Z ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc No_Coverage_Flux For Solid_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

Species

Constant

Syntax

Bc No_Coverage_Flux For Species ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | *{flux | value}* = *flux*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
flux	real	–

Exponential

Syntax

Bc No_Coverage_Flux For Species ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Free_Open_Flow

Syntax

Bc No_Coverage_Flux For Species ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Free_Open_Flow [Toggle = *toggle*]

Summary

This Flux BC sets the surface normal species flux equal to the diffusive flux.

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–

Global

Syntax

Bc No_Coverage_Flux For Species ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
global_name	“string”	–
component	integer	0

Interface

Syntax

Bc No_Coverage_Flux For Species ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Interface [Toggle = *toggle*]

Summary

Flux BC model of species for moving interfaces

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–

Nat_Conv

Syntax

Bc No_Coverage_Flux For Species ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nat_Conv ([Toggle = *toggle*] | K = *k* | Yinf = *yinf*)

Summary

natconv species flux

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
k	real	–
yinf	real	–

Nd_Table

Syntax

Bc No_Coverage_Flux For Species ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Open_Adv_Flow

Syntax

Bc No_Coverage_Flux For Species ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Open_Adv_Flow ([Toggle = *toggle*] | [Far_Field_Entrainment_Value = *far_field_entrainment_value*])

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
far_field_entrainment_value	real	0

Polynomial

Syntax

Bc No_Coverage_Flux For Species ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc No_Coverage_Flux For Species ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f	"string"	–

User_Field

Syntax

Bc No_Coverage_Flux For Species ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc No_Coverage_Flux For Species ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc No_Coverage_Flux For Species ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

Voltage

Constant

Syntax

Bc No_Coverage_Flux For Voltage ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Toggle = *toggle*] | *{flux | value}* = *flux*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
flux	real	–

Exponential

Syntax

Bc No_Coverage_Flux For Voltage ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc No_Coverage_Flux For Voltage ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Bc No_Coverage_Flux For Voltage ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc No_Coverage_Flux For Voltage ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc No_Coverage_Flux For Voltage ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
f	"string"	–

User_Field

Syntax

Bc No_Coverage_Flux For Voltage ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc No_Coverage_Flux For Voltage ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Bc No_Coverage_Flux For Voltage ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
toggle	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

Bc Rotated

Cvfem_Mesh

Constant

Syntax

Bc Rotated For Cvfem_Mesh ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Unrotated = *unrotated*] | [Valuex = *valuex*] | [Valuey = *valuey*] | [Valuez = *valuez*])

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
unrotated	"string"	–
valuex	real	0
valuey	real	0
valuez	real	0

Kinematic

Syntax

Bc Rotated For Cvfem_Mesh ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Kinematic ([Unrotated = *unrotated*] | [V0 = *v0*] | [Shared_Normal = *shared_normal*] | [No_Mesh_Movement = *no_mesh_movement*])

Summary

Kinematic model for rotated CVFEM_MESH boundary condition

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
unrotated	“string”	–
v0	real	0
shared_normal	“string”	–
no_mesh_movement	“string”	–

Pyrolysis

Syntax

Bc Rotated For Cvfem_Mesh ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Pyrolysis ([Unrotated = *unrotated*] | [T_Ref = *t_ref*] | K = *k* | K_0 = *k_0* | N = *n* | [Temperature = *temperature*])

Summary

Pyrolysis model for rotated CVFEM_MESH boundary condition

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
unrotated	“string”	–
t_ref	real	0
k	real	–
k_0	real	–
n	real	–
temperature	“string”	–

Recession

Syntax

Bc Rotated For Cvfem_Mesh ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Recession ([Unrotated = *unrotated*] | [P_Ref = *p_ref*] | K = *k* | K_0 = *k_0* | N = *n* | [Pressure = *pressure*])

Summary

Recession model for rotated CVFEM_MESH boundary condition

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
unrotated	"string"	–
p_ref	real	0
k	real	–
k_0	real	–
n	real	–
pressure	"string"	–

Vector_String_Function

Syntax

Bc Rotated For Cvfem_Mesh ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Vector_String_Function ([Unrotated = *unrotated*] | F_X = *f_x* | F_Y = *f_y* | [F_Z = *f_z*])

Summary

Value from a string function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
unrotated	"string"	–
f_x	"string"	–
f_y	"string"	–
f_z	"string"	–

Cvfem_Mesh_X

Constant

Syntax

Bc Rotated For Cvfem_Mesh_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Unrotated = *unrotated*] | Value = *value*)

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
unrotated	“string”	–
value	real	–

Exponential

Syntax

Bc Rotated For Cvfem_Mesh_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Unrotated = *unrotated*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
unrotated	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Rotated For Cvfem_Mesh_X ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Unrotated = *unrotated*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
unrotated	"string"	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Bc Rotated For Cvfem_Mesh_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Unrotated = *unrotated*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
unrotated	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc Rotated For Cvfem_Mesh_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Unrotated = *unrotated*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
unrotated	"string"	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Rotated For Cvfem_Mesh_X ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Unrotated = *unrotated*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name.

Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
unrotated	“string”	–
f	“string”	–

User_Field

Syntax

Bc Rotated For Cvfem_Mesh_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Unrotated = *unrotated*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
unrotated	“string”	–
name	“string”	–
scaling	real	1
global_var	“string”	–
time_function	“string”	–

User_Function

Syntax

Bc Rotated For Cvfem_Mesh_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Unrotated = *unrotated*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
unrotated	“string”	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Bc Rotated For Cvfem_Mesh_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Unrotated = *unrotated*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
unrotated	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

Cvfem_Mesh_Y

Constant

Syntax

Bc Rotated For Cvfem_Mesh_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Unrotated = *unrotated*] | Value = *value*)

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
unrotated	"string"	–
value	real	–

Exponential

Syntax

Bc Rotated For Cvfem_Mesh_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Unrotated = *unrotated*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
unrotated	“string”	–
variable	“string”	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Rotated For Cvfem_Mesh_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Unrotated = *unrotated*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
unrotated	“string”	–
global_name	“string”	–
component	integer	0

Nd_Table

Syntax

Bc Rotated For Cvfem_Mesh_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Unrotated = *unrotated*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
unrotated	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc Rotated For Cvfem_Mesh_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Unrotated = *unrotated*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
unrotated	"string"	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Rotated For Cvfem_Mesh_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Unrotated = *unrotated*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
unrotated	“string”	–
f	“string”	–

User_Field

Syntax

Bc Rotated For Cvfem_Mesh_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Unrotated = *unrotated*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
unrotated	“string”	–
name	“string”	–
scaling	real	1
global_var	“string”	–
time_function	“string”	–

User_Function

Syntax

Bc Rotated For Cvfem_Mesh_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Unrotated = *unrotated*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
unrotated	“string”	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Bc Rotated For Cvfem_Mesh_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Unrotated = *unrotated*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
unrotated	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

Cvfem_Mesh_Z

Constant

Syntax

Bc Rotated For Cvfem_Mesh_Z ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Unrotated = *unrotated*] | Value = *value*)

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
unrotated	"string"	–
value	real	–

Exponential

Syntax

Bc Rotated For Cvfem_Mesh_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Unrotated = *unrotated*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
unrotated	“string”	–
variable	“string”	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Rotated For Cvfem_Mesh_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Unrotated = *unrotated*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
unrotated	"string"	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Bc Rotated For Cvfem_Mesh_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Unrotated = *unrotated*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
unrotated	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc Rotated For Cvfem_Mesh_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Unrotated = *unrotated*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
unrotated	"string"	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Rotated For Cvfem_Mesh_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Unrotated = *unrotated*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
unrotated	"string"	–
f	"string"	–

User_Field

Syntax

Bc Rotated For Cvfem_Mesh_Z ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Unrotated = *unrotated*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
unrotated	"string"	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Bc Rotated For Cvfem_Mesh_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Unrotated = *unrotated*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
unrotated	“string”	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Bc Rotated For Cvfem_Mesh_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Unrotated = *unrotated*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
unrotated	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

Cvfem_Momentum

Constant

Syntax

Bc Rotated For Cvfem_Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Unrotated = *unrotated*] | [Valuex = *valuex*] | [Valuey = *valuey*] | [Valuez = *valuez*])

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
unrotated	"string"	–
valuex	real	0
valuey	real	0
valuez	real	0

Kinematic

Syntax

Bc Rotated For Cvfem_Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Kinematic ([Unrotated = *unrotated*] | [V0 = *v0*] | [Shared_Normal = *shared_normal*] | [No_Mesh_Movement = *no_mesh_movement*])

Summary

Kinematic model for rotated cvfem momentum

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
unrotated	“string”	–
v0	real	0
shared_normal	“string”	–
no_mesh_movement	“string”	–

Vector_String_Function

Syntax

Bc Rotated For Cvfem_Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Vector_String_Function ([Unrotated = *unrotated*] | F_X = *f_x* | F_Y = *f_y* | [F_Z = *f_z*])

Summary

Value from a string function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
unrotated	“string”	–
f_x	“string”	–
f_y	“string”	–
f_z	“string”	–

Cvfem_Momentum_X

Constant

Syntax

Bc Rotated For Cvfem_Momentum_X ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Unrotated = *unrotated*] | Value = *value*)

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
unrotated	“string”	–
value	real	–

Exponential

Syntax

Bc Rotated For Cvfem_Momentum_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Unrotated = *unrotated*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
unrotated	“string”	–
variable	“string”	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Rotated For Cvfem_Momentum_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Unrotated = *unrotated*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
unrotated	"string"	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Bc Rotated For Cvfem_Momentum_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Unrotated = *unrotated*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
unrotated	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc Rotated For Cvfem_Momentum_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Unrotated = *unrotated*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
unrotated	"string"	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Rotated For Cvfem_Momentum_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Unrotated = *unrotated*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
unrotated	“string”	–
f	“string”	–

User_Field

Syntax

Bc Rotated For Cvfem_Momentum_X ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Unrotated = *unrotated*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
unrotated	“string”	–
name	“string”	–
scaling	real	1
global_var	“string”	–
time_function	“string”	–

User_Function

Syntax

Bc Rotated For Cvfem_Momentum_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Unrotated = *unrotated*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
unrotated	“string”	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Bc Rotated For Cvfem_Momentum_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Unrotated = *unrotated*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
unrotated	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

Cvfem_Momentum_Y

Constant

Syntax

Bc Rotated For Cvfem_Momentum_Y ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Unrotated = *unrotated*] | Value = *value*)

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
unrotated	"string"	–
value	real	–

Exponential

Syntax

Bc Rotated For Cvfem_Momentum_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Unrotated = *unrotated*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
unrotated	“string”	–
variable	“string”	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Rotated For Cvfem_Momentum_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Unrotated = *unrotated*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
unrotated	"string"	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Bc Rotated For Cvfem_Momentum_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Unrotated = *unrotated*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
unrotated	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc Rotated For Cvfem_Momentum_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Unrotated = *unrotated*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
unrotated	"string"	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Rotated For Cvfem_Momentum_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Unrotated = *unrotated*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
unrotated	“string”	–
f	“string”	–

User_Field

Syntax

Bc Rotated For Cvfem_Momentum_Y ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Unrotated = *unrotated*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
unrotated	“string”	–
name	“string”	–
scaling	real	1
global_var	“string”	–
time_function	“string”	–

User_Function

Syntax

Bc Rotated For Cvfem_Momentum_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Unrotated = *unrotated*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
unrotated	“string”	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Bc Rotated For Cvfem_Momentum_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Unrotated = *unrotated*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
unrotated	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

Cvfem_Momentum_Z

Constant

Syntax

Bc Rotated For Cvfem_Momentum_Z ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Unrotated = *unrotated*] | Value = *value*)

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
unrotated	"string"	–
value	real	–

Exponential

Syntax

Bc Rotated For Cvfem_Momentum_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Unrotated = *unrotated*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
unrotated	“string”	–
variable	“string”	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Rotated For Cvfem_Momentum_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Unrotated = *unrotated*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
unrotated	"string"	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Bc Rotated For Cvfem_Momentum_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Unrotated = *unrotated*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
unrotated	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc Rotated For Cvfem_Momentum_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Unrotated = *unrotated*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
unrotated	"string"	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Rotated For Cvfem_Momentum_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Unrotated = *unrotated*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
unrotated	“string”	–
f	“string”	–

User_Field

Syntax

Bc Rotated For Cvfem_Momentum_Z ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Unrotated = *unrotated*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
unrotated	“string”	–
name	“string”	–
scaling	real	1
global_var	“string”	–
time_function	“string”	–

User_Function

Syntax

Bc Rotated For Cvfem_Momentum_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Unrotated = *unrotated*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
unrotated	“string”	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Bc Rotated For Cvfem_Momentum_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Unrotated = *unrotated*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
unrotated	“string”	–
Name	“string”	–
plugin_parameters	“string”	–

Mesh

Constant

Syntax

Bc Rotated For Mesh ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Unrotated = *unrotated*] | Value = *value*)

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
unrotated	“string”	–
value	real	–

Exponential

Syntax

Bc Rotated For Mesh ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Unrotated = *unrotated*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
unrotated	“string”	–
variable	“string”	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Rotated For Mesh ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Unrotated = *unrotated*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
unrotated	“string”	–
global_name	“string”	–
component	integer	0

Kinematic

Syntax

Bc Rotated For Mesh ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Kinematic ([Unrotated = *unrotated*] | [V0 = *v0*] | [Shared_Normal = *shared_normal*] | [No_Mesh_Movement = *no_mesh_movement*])

Summary

Kinematic model for rotated mesh boundary condition

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
unrotated	“string”	–
v0	real	0
shared_normal	“string”	–
no_mesh_movement	“string”	–

Nd_Table

Syntax

Bc Rotated For Mesh ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Unrotated = *unrotated*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
unrotated	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc Rotated For Mesh ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Unrotated = *unrotated*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
unrotated	“string”	–
variable	“string”	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Pyrolysis

Syntax

Bc Rotated For Mesh ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Pyrolysis ([Unrotated = *unrotated*] | [T_Ref = *t_ref*] | K = *k* | K_0 = *k_0* | N = *n* | [Temperature = *temperature*])

Summary

Pyrolysis model for rotated mesh boundary condition

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
unrotated	“string”	–
t_ref	real	0
k	real	–
k_0	real	–
n	real	–
temperature	“string”	–

Recession

Syntax

Bc Rotated For Mesh ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Recession ([Unrotated = *unrotated*] | [P_Ref = *p_ref*] | K = *k* | K_0 = *k_0* | N = *n* | [Pressure = *pressure*])

Summary

Recession model for rotated mesh boundary condition

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
unrotated	"string"	–
p_ref	real	0
k	real	–
k_0	real	–
n	real	–
pressure	"string"	–

Scalar_String_Function

Syntax

Bc Rotated For Mesh ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Unrotated = *unrotated*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
unrotated	“string”	–
f	“string”	–

User_Field

Syntax

Bc Rotated For Mesh ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Unrotated = *unrotated*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
unrotated	“string”	–
name	“string”	–
scaling	real	1
global_var	“string”	–
time_function	“string”	–

User_Function

Syntax

Bc Rotated For Mesh ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Unrotated = *unrotated*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
unrotated	“string”	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Bc Rotated For Mesh ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Unrotated = *unrotated*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
unrotated	“string”	–
Name	“string”	–
plugin_parameters	“string”	–

Momentum

Constant

Syntax

Bc Rotated For Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Unrotated = *unrotated*] | [Valuex = *valuex*] | [Valuey = *valuey*] | [Valuez = *valuez*])

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
unrotated	“string”	–
valuex	real	0
valuey	real	0
valuez	real	0

Kinematic

Syntax

Bc Rotated For Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Kinematic ([Unrotated = *unrotated*] | [V0 = *v0*] | [Shared_Normal = *shared_normal*] | [No_Mesh_Movement = *no_mesh_movement*])

Summary

Kinematic model for rotated momentum

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
unrotated	“string”	–
v0	real	0
shared_normal	“string”	–
no_mesh_movement	“string”	–

Vector_String_Function

Syntax

Bc Rotated For Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Vector_String_Function ([Unrotated = *unrotated*] | F_X = *f_x* | F_Y = *f_y* | [F_Z = *f_z*])

Summary

Value from a string function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
unrotated	“string”	–
f_x	“string”	–
f_y	“string”	–
f_z	“string”	–

Momentum_X

Constant

Syntax

Bc Rotated For Momentum_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Unrotated = *unrotated*] | Value = *value*)

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
unrotated	“string”	–
value	real	–

Exponential

Syntax

Bc Rotated For Momentum_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Unrotated = *unrotated*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
unrotated	“string”	–
variable	“string”	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Rotated For Momentum_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Unrotated = *unrotated*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
unrotated	"string"	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Bc Rotated For Momentum_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Unrotated = *unrotated*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
unrotated	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc Rotated For Momentum_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Unrotated = *unrotated*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
unrotated	"string"	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Rotated For Momentum_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Unrotated = *unrotated*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
unrotated	“string”	–
f	“string”	–

User_Field

Syntax

Bc Rotated For Momentum_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Unrotated = *unrotated*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
unrotated	“string”	–
name	“string”	–
scaling	real	1
global_var	“string”	–
time_function	“string”	–

User_Function

Syntax

Bc Rotated For Momentum_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Unrotated = *unrotated*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
unrotated	“string”	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Bc Rotated For Momentum_X ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Unrotated = *unrotated*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
unrotated	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

Momentum_Y

Constant

Syntax

Bc Rotated For Momentum_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Unrotated = *unrotated*] | Value = *value*)

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
unrotated	"string"	–
value	real	–

Exponential

Syntax

Bc Rotated For Momentum_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Unrotated = *unrotated*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
unrotated	“string”	–
variable	“string”	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Rotated For Momentum_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Unrotated = *unrotated*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
unrotated	“string”	–
global_name	“string”	–
component	integer	0

Nd_Table

Syntax

Bc Rotated For Momentum_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Unrotated = *unrotated*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
unrotated	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc Rotated For Momentum_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Unrotated = *unrotated*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
unrotated	"string"	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Rotated For Momentum_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Unrotated = *unrotated*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
unrotated	“string”	–
f	“string”	–

User_Field

Syntax

Bc Rotated For Momentum_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Unrotated = *unrotated*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
unrotated	“string”	–
name	“string”	–
scaling	real	1
global_var	“string”	–
time_function	“string”	–

User_Function

Syntax

Bc Rotated For Momentum_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Unrotated = *unrotated*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
unrotated	“string”	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Bc Rotated For Momentum_Y ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Unrotated = *unrotated*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
unrotated	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

Momentum_Z

Constant

Syntax

Bc Rotated For Momentum_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Constant ([Unrotated = *unrotated*] | Value = *value*)

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
unrotated	"string"	–
value	real	–

Exponential

Syntax

Bc Rotated For Momentum_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Exponential ([Unrotated = *unrotated*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
unrotated	“string”	–
variable	“string”	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Bc Rotated For Momentum_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Global ([Unrotated = *unrotated*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
unrotated	“string”	–
global_name	“string”	–
component	integer	0

Nd_Table

Syntax

Bc Rotated For Momentum_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Nd_Table ([Unrotated = *unrotated*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
unrotated	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Bc Rotated For Momentum_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Polynomial ([Unrotated = *unrotated*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
unrotated	“string”	–
variable	“string”	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Bc Rotated For Momentum_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Scalar_String_Function ([Unrotated = *unrotated*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
unrotated	“string”	–
f	“string”	–

User_Field

Syntax

Bc Rotated For Momentum_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Field ([Unrotated = *unrotated*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
unrotated	“string”	–
name	“string”	–
scaling	real	1
global_var	“string”	–
time_function	“string”	–

User_Function

Syntax

Bc Rotated For Momentum_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Function ([Unrotated = *unrotated*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
unrotated	“string”	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Bc Rotated For Momentum_Z ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = User_Plugin ([Unrotated = *unrotated*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
unrotated	“string”	–
Name	“string”	–
plugin_parameters	“string”	–

Bc Uncoupled_Bulknode_Flux

Porous_Enthalpy

Bulk_Node_Open_Uncoupled

Syntax

Bc Uncoupled_Bulknode_Flux For Porous_Enthalpy ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) = Bulk_Node_Open_Uncoupled Bulk_Node = *bulk_node*

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
bulk_node	“string”	–

Periodic BCs

Periodic

Scope

Aria Region

Summary

The periodic command block is used to define periodic, or cyclic, boundary conditions.

Description

For standard periodic boundary conditions, the line commands SOURCE, COPY, SEARCH TOLERANCE should appear, while the line commands THETA, POINT ON AXIS, and REFERENCE AXIS must not appear.

Cyclic boundary conditions, the line commands SOURCE, COPY, SEARCH TOLERANCE, THETA, POINT ON AXIS, and REFERENCE AXIS must appear.

Two self-similar side sets for the boundary condition are specified by the SOURCE and COPY command lines. It must be possible to replicate the coordinates for the nodes in one set by a simple translation of the corresponding nodes in the other set.

A geometric check is made to determine if the side sets are self similar. The geometric check is set to a certain tolerance. This tolerance must be set to a small positive number (e.g. 10^{-4}) by using the command line SEARCH TOLERANCE. After translation each node in the set of copy nodes lies within a radius defined by the search tolerance from a node in the set of master nodes. Additionally, upon transformation, copy nodes must lie in the interpolation space of at least one facet in the master surface. When these two conditions are satisfied the two sets are considered self-similar.

```
begin Periodic Name

   Copy {=} name

   Master {=} Master

   Point On Axis {=} Point

   Reference Axis {=} Axis

   Search Tolerance {=} SearchTolerance

   Slave {=} Slave
```

```
   Source {=} name

   Theta {=} Theta

end Periodic Name
```

Line Commands

Copy

Syntax

Copy *{=} name*

Summary

Defines copy surface for this interaction. When used in Aria this must be a sideset.

Parameter	Value	Default
{=}	{= | are | is}	–
name	string	–

Master

Syntax

Master *{=} Master*

Summary

Defines master surface for this interaction. When used in Aria this must always be a sideset.

Parameter	Value	Default
{=}	{= | are | is}	–
Master	string	–

Point On Axis

Syntax

Point On Axis *{=} Point*

Summary

This line command defines a point that lies on the reference axis.

Parameter	Value	Default
{=}	{= | are | is}	–
Point	string	–

Reference Axis

Syntax

Reference Axis *{=}* *Axis*

Summary

This line command defines the reference wedge axis for the periodic BC about which a source surface is rotated to transform it into the copy surface.

Parameter	Value	Default
{=}	{= | are | is}	–
Axis	string	–

Search Tolerance

Syntax

Search Tolerance *{=}* *SearchTolerance*

Summary

This line command defines the search tolerance used to determine if two node sets are self similar.

Parameter	Value	Default
{=}	{= | are | is}	–
SearchTolerance	real	–

Slave

Syntax

Slave *{=}* *Slave*

Summary

Defines slave surface for this interaction. When used in Aria, this must be a sideset.

Parameter	Value	Default
{=}	{= | are | is}	–
Slave	string	–

Source

Syntax

Source *{=}* *name*

Summary

Defines source surface for this interaction. When used in Aria, this must always be a sideset.

Parameter	Value	Default
{=}	{= | are | is}	–
name	string	–

Theta

Syntax

Theta *{=} Theta*

Summary

This line command defines the angle, in degrees, for a periodic boundary condition. A positive angle is defined by sweeping the source surface through the domain to the copy surface about the reference axis.

Parameter	Value	Default
{=}	{= | are | is}	–
Theta	real	–

Define Direction

Syntax

Define Direction *DirectName* With Vector *Components1 Components2 Components3*

Summary

Defines a named spatial direction in terms of vector components.

Parameter	Value	Default
DirectName	string	–
Components	real1 real2 real3	–

Define Point

Syntax

Define Point *PointName* With Coordinates *Coordinates1 Coordinates2 Coordinates3*

Summary

Defines a named point in space in terms of its coordinates.

Parameter	Value	Default
PointName	string	–
Coordinates	real1 real2 real3	–

Legacy BC Models

Aerodynamic Convection Heat Transfer

High-speed flows about bodies are often characterized by large temperature gradients and large variations of the flow properties through the boundary layer. The problem of heat transfer about the body is often addressed using numerical techniques of CFD to resolve the thermal transport through the boundary layer. Because CFD resources may not be readily applied at the design stage researchers often apply simplifying assumptions to arrive at alternative models for solving the problem of heat transfer about bodies exposed to high-speed flow. One common modeling approach is to define a convection coefficient and reference temperature that depend upon flight conditions. The aero heat flux boundary condition provides a means for supplying details of the flight conditions to arrive at a representative convection coefficient and reference temperature for two heat flux models, one derived by [58] and one that varies only with altitude.

The model by Eckert suggests that a convective heat transfer approach ignoring boundary layer temperature gradients may be used if the properties are evaluated at an alternative reference temperature representative of the averaged flow conditions. Syntax for typical usage of this model is shown below.

```
Begin AERO HEAT FLUX BOUNDARY CONDITION body_heating
  START TIME =  0.0
  STOP TIME =  20.0
  freestream temperature = 233.15
  freestream pressure    = 5.066e3
  mach number = 3.0
  fluid viscosity temperature function            = air_viscosity
  fluid specific heat temperature function        = air_specheat
  fluid thermal conductivity temperature function = air_cond
  fluid gamma = 1.4
  adiabatic wall temperature from recovery factor
  eckert convective coefficient
  COORDINATE OFFSET AXIS = x
  cone length = 0.0
  add surface surface_1
  integrated power output total_heating
End
```

Another aerodynamic heat flux model defines a convective coefficient that relies upon the variation of density with altitude. Syntax for typical usage of this model is shown below.

```
Begin AERO HEAT FLUX BOUNDARY CONDITION body_heating
  START TIME =  0.0
  STOP TIME =  20.0
```

(continues on next page)

(continued from previous page)

```
  Density Ratio Convective Coefficient
  Freestream Density Altitude Function = density_func
  Altitude time function = altitude_time_func
  Reference Density = 1.15
  Reference HTC = 25.0
  Density Ratio exponent = 0.6
  reference temperature time function = ref_temp_func
  add surface surface_1
  integrated power output total_heating
End
```

A more complete description of syntax for the aero heating models is described below

Aero Heat Flux Boundary Condition

Scope

Aria Region, Equation System, Explicit Equation System, Root Finder Equation System

Summary

This command block defines a distributed convective heat flux about a body based upon aerodynamic flight conditions applied on the given surface.

Description

The aerodynamic heat flux depends upon characterization of flight conditions prescribed using data tables and spatial location on the body of interest.

```
begin Aero Heat Flux Boundary Condition Name

   Add Surface SurfaceList...

   Adiabatic Wall Temperature {=} Value

   Adiabatic Wall Temperature Altitude Function {=} Function

   Adiabatic Wall Temperature From Recovery Factor

   Adiabatic Wall Temperature Time Function {=} Function

   Altitude {=} Value

   Altitude Time Function {=} Function

   Cone Length {=} Value
```

Coordinate Offset {=} *Value*

Coordinate Offset Axis {=} *axis*

Density Ratio Convective Coefficient

Density Ratio Exponent {=} *Value*

Eckert Convective Coefficient

Fluid Gamma {=} *Value*

Fluid Gas Constant {=} *Value*

Fluid Properties Temperature Function {=} *Function*

Fluid Specific Heat Temperature Function {=} *Function*

Fluid Thermal Conductivity Temperature Function {=} *Function*

Fluid Viscosity Temperature Function {=} *Function*

Freestream Density Altitude Function {=} *Function*

Freestream Pressure {=} *Value*

Freestream Pressure Altitude Function {=} *Function*

Freestream Pressure Time Function {=} *Function*

Freestream Temperature {=} *Value*

Freestream Temperature Altitude Function {=} *Function*

Freestream Temperature Time Function {=} *Function*

Integrated Flux Output *VariableName*

Integrated Power Output *VariableName*

Mach Number {=} *Value*

Mach Number Time Function {=} *Function*

```
   Reference Density {=} Value

   Reference Htc {=} Value

   Reference Temperature {=} Value

   Reference Temperature Time Function {=} FunctionName

   Start Time {=} Value

   Stop Time {=} Value

   Uq Flux Multiplier {=} Value

   Use Toggle Block ToggleName [ {on} ElementBlockList...  ]

end Aero Heat Flux Boundary Condition Name
```

Line Commands

Add Surface

Syntax

Add Surface *SurfaceList...*

Summary

Adds surfaces, by name, to a boundary condition's extent.

Description

This line command is used to add surfaces to the extent of a boundary condition. In Exodus II, surfaces are specified as side sets, that have a global integer identifier. For example, side set 12 would be added by this line command using the surface name surface_12. Note that in SIERRA, each element of an array of strings must be separated by whitespace.

Parameter	Value	Default
SurfaceList	string...	–

Adiabatic Wall Temperature

Syntax

Adiabatic Wall Temperature *{=} Value*

Summary

Specifies a constant adiabatic wall temperature. Usage: ECKERT CONVECTIVE COEFFICIENT.

FREESTREAM TEMPERATURE and fluid properties consistent with this ADIABATIC WALL TEMPERATURE must also be provided.

Parameter	Value	Default
{=}	{= | are | is}	–
Value	real	0.0

Adiabatic Wall Temperature Altitude Function

Syntax

Adiabatic Wall Temperature Altitude Function *{=}* *Function*

Summary

Specifies the adiabatic wall temperature versus altitude function for climate condition. Usage: ECKERT CONVECTIVE COEFFICIENT.

FREESTREAM TEMPERATURE and fluid properties consistent with this ADIABATIC WALL TEMPERATURE must also be provided.

Parameter	Value	Default
{=}	{= | are | is}	–
Function	string	–

Adiabatic Wall Temperature From Recovery Factor

Syntax

Adiabatic Wall Temperature From Recovery Factor

Summary

Specifies that the adiabatic wall temperature be computed using the recovery factor, the stagnation temperature and the freestream temperature. Usage: ECKERT CONVECTIVE COEFFICIENT.

Description

The adiabatic wall temperature T_{aw} is computed as

$$T_{aw} = rT_o + (1 - r)T_\infty$$

where r is the recovery factor, T_o is the stagnation temperature and T_∞ is the freestream temperature. The user must provide specification of MACH NUMBER, FREESTREAM TEMPERATURE, FREESTREAM PRESSURE and FLUID_GAMMA models. Stagnation temperature and flow velocity v will be internally computed from the MACH NUMBER. The recovery factor is given in terms of the freestream Prandtl number $C_p\mu/K$ as

$$r = \begin{cases} Pr^{1/2} & \text{laminar flow} \\ Pr^{1/3} & \text{turbulent flow} \quad Re > 5X10^5 \end{cases} .$$

Hence one must provide functions for specific heat, dynamic viscosity and thermal conductivity as a function of temperature. The flow regime varies spatially in the COORDINATE OFFSET direction as determined using the local Reynolds number

$$Re_x = \frac{\rho v x}{\mu} .$$

Here the pressure P will vary with altitude hence density is evaluated using the ideal gas law

$$\rho = \frac{P}{RT_\infty}$$

Adiabatic Wall Temperature Time Function

Syntax

Adiabatic Wall Temperature Time Function *{=}* *Function*

Summary

Specifies the name of adiabatic wall temperature versus time function. Usage: ECKERT CONVECTIVE COEFFICIENT.

FREESTREAM TEMPERATURE and fluid properties consistent with this ADIABATIC WALL TEMPERATURE must also be provided.

Parameter	Value	Default
{=}	{= | are | is}	–
Function	string	–

Altitude

Syntax

Altitude *{=}* *Value*

Summary

Specifies a constant altitude.

Used in conjunction with FREESTREAM TEMPERATURE ALTITUDE FUNCTION, FREESTREAM TEMPERATURE FUNCTION and ECKERT CONVECTIVE COEFFICIENT. Alternative usage with FREESTREAM DENSITY ALTITUDE FUNCTION and DENSITY RATIO CONVECTIVE COEFFICIENT.

Parameter	Value	Default
{=}	{= | are | is}	–
Value	real	0.0

Altitude Time Function

Syntax

Altitude Time Function *{=}* *Function*

Summary

Specifies the altitude versus time function for a given flight. The tabulated time is defined relative to the START TIME.

Used in conjunction with FREESTREAM TEMPERATURE ALTITUDE FUNCTION, FREESTREAM TEMPERATURE FUNCTION and ECKERT CONVECTIVE COEFFICIENT. Alternative usage with FREESTREAM DENSITY ALTITUDE FUNCTION and DENSITY RATIO CONVECTIVE COEFFICIENT.

Parameter	Value	Default
{=}	{= | are | is}	–
Function	string	–

Cone Length

Syntax

Cone Length *{=}* *Value*

Summary

Specifies the length at which the body profile transitions to a constant diameter. Usage: ECKERT CONVECTIVE COEFFICIENT.

Parameter	Value	Default
{=}	{= | are | is}	–
Value	real	0.0

Coordinate Offset

Syntax

Coordinate Offset *{=}* *Value*

Summary

Specifies the coordinate offset from which the spatial position will be defined in the ECKERT CONVECTIVE COEFFICIENT calculations $x = X -$ offset where X is the Cartesian model coordinate from the mesh prescribed using COORDINATE OFFSET AXIS and is assumed to define the leading edge. The value of x is used in computation of the Reynolds number $Re_x^* = \frac{\rho^* v x}{\mu}$ hence the evaluation of x must provide values greater than zero.

Parameter	Value	Default
{=}	{= | are | is}	–
Value	real	0.0

Coordinate Offset Axis

Syntax

Coordinate Offset Axis *{=}* *axis*

Summary

Specifies the Cartesian coordinate axis (X, Y or Z) from which spatial position will be defined when using the ECKERT CONVECTIVE COEFFICIENT. The surface heat transfer coefficient will be computed spatially relative to zero on this axis.

For models in which the geometry is offset from a zero reference one must define the COORDINATE OFFSET.

Parameter	Value	Default
{=}	{= | are | is}	–
axis	string	none

Density Ratio Convective Coefficient

Syntax

Density Ratio Convective Coefficient

Summary

Specifies a density ratio heat transfer coefficient model. Requires that FREESTREAM DENSITY ALTITUDE FUNCTION, REFERENCE DENSITY, REFERENCE HTC, DENSITY RATIO EXPONENT, START TIME and STOP TIME specifications be supplied.

Description

The heat transfer coefficient that varies as

$$h = h_o \left[\frac{\rho(A)}{\rho_o} \right]^n$$

where h_o is the reference heat transfer coefficient, $\rho(A)$ is the altitude dependent density, ρ_o is the reference density and n is the density ratio exponent. Here the model parameters h_o, ρ_o and n are calibrated based upon measured temperature data.

Density Ratio Exponent

Syntax

Density Ratio Exponent *{=}* *Value*

Summary

Specifies the density ratio exponent value for the density ratio heat transfer coefficient model. Usage: DENSITY RATIO CONVECTIVE COEFFICIENT.

Parameter	Value	Default
{=}	{= | are | is}	–
Value	real	0.0

Eckert Convective Coefficient

Syntax

Eckert Convective Coefficient

Summary

Specifies use of the Eckert heat transfer coefficient model for high-speed flow. Requires that freestream FLUID PROPERTY TEMPERATURE FUNCTIONs, START TIME, STOP TIME, FREESTREAM PRESSURE and ADIABATIC WALL TEMPERATURE specifications be supplied.

Description

A constant property heat transfer coefficient approach in which properties are evaluated at an alternative temperature

$$T^* = T_\infty + 0.5(T - T_\infty) + 0.22(T_{aw} - T_\infty)$$

where T_∞ is the freestream temperature, T is the surface temperature, T_{aw} is the adiabatic wall temperature. T_{aw} will vary spatially depending upon the local Reynolds number

$$Re_x^* = \frac{\rho^* v x}{\mu}$$

where μ is evaluated at T^* and the coordinate $x > 0$. The FREESTREAM PRESSURE varies with altitude hence the density ρ^* is evaluated using the ideal gas law

$$\rho^* = \frac{P}{RT^*} .$$

The Stanton number represents the ratio of heat transferred into the fluid flow to the thermal capacity of the fluid. By developing the relationship between fluid friction and heat transfer frictional resistance, the heat transfer can be expressed in terms of the Stanton number to arrive at an approximate heat transfer coefficient for various portions of the flow regime

$$St_x^* Pr^{*2/3} = \begin{cases} 0.332(Re_x^*/f_M)^{-1/2} & Re_x^* < 5 \times 10^5 \text{ laminar} \\ 0.0296(Re_x^*/f_M)^{-1/5} & 5 \times 10^5 < Re_x^* < 10^7 \text{ turbulent} \\ 0.185[log10(Re_x^*/f_M)]^{-2.584} & 10^7 < Re_x^* < 10^9 \text{ turbulent} \\ 0.00063 & Re_x^* > 10^9 \text{ turbulent} \end{cases}$$

The parameter f_M is the Mangler transformation for cone geometry

$$f_M = \begin{cases} 3 & \text{cone geometry, laminar} \\ 2 & \text{cone geometry, turbulent} \\ 1 & \text{flat plate geometry, laminar} \end{cases}$$

While Eckert's approach considers an average heat transfer coefficient over portions of a surface, here we consider a local heat transfer coefficient, $h(x)$ based upon the above relations that can be applied directly to the surface discretization

$$h(x) = \rho^* C_p^* U_\infty St_x^*$$

Fluid Gamma

Syntax

Fluid Gamma *{=}* *Value*

Summary

Specifies a constant specific heat ratio of the fluid medium. Usage: ECKERT CONVECTIVE COEFFICIENT.

Only constant values of γ are allowed.

Parameter	Value	Default
{=}	{= | are | is}	–
Value	real	1.4

Fluid Gas Constant

Syntax

Fluid Gas Constant *{=}* *Value*

Summary

Specifies the gas constant of the fluid medium. Usage: ECKERT CONVECTIVE COEFFICIENT.

Parameter	Value	Default
{=}	{= | are | is}	–
Value	real	none

Fluid Properties Temperature Function

Syntax

Fluid Properties Temperature Function *{=}* *Function*

Summary

Specifies a multi-column function containing fluid properties density, specific heat, conductivity and dynamic viscosity versus temperature with respective property columns named CP, K and MU. Usage: ECKERT CONVECTIVE COEFFICIENT.

Parameter	Value	Default
{=}	{= | are | is}	–
Function	string	–

Fluid Specific Heat Temperature Function

Syntax

Fluid Specific Heat Temperature Function *{=}* *Function*

Summary

Specifies the name of a user tabular function for specific heat as a function of temperature. Usage: ECKERT CONVECTIVE COEFFICIENT.

Parameter	Value	Default
{=}	{= | are | is}	–
Function	string	–

Fluid Thermal Conductivity Temperature Function

Syntax

Fluid Thermal Conductivity Temperature Function *{=}* *Function*

Summary

Specifies the name of a user tabular function for thermal conductivity as a function of temperature. Usage: ECKERT CONVECTIVE COEFFICIENT.

Parameter	Value	Default
{=}	{= | are | is}	–
Function	string	–

Fluid Viscosity Temperature Function

Syntax

Fluid Viscosity Temperature Function *{=}* *Function*

Summary

Specifies the name of a user tabular function for dynamic viscosity as a function of temperature. Usage: ECKERT CONVECTIVE COEFFICIENT.

Parameter	Value	Default
{=}	{= | are | is}	–
Function	string	–

Freestream Density Altitude Function

Syntax

Freestream Density Altitude Function *{=}* *Function*

Summary

Specifies the freestream density versus altitude function for climate condition. Usage: DENSITY RATIO CONVECTIVE COEFFICIENT.

Requires that one supply an ALTITUDE model.

Parameter	Value	Default
{=}	{= | are | is}	–
Function	string	–

Freestream Pressure

Syntax

Freestream Pressure *{=}* *Value*

Summary

Specifies a constant freestream pressure.

Parameter	Value	Default
{=}	{= | are | is}	–
Value	real	0.0

Freestream Pressure Altitude Function

Syntax

Freestream Pressure Altitude Function *{=}* *Function*

Summary

Specifies the freestream pressure versus altitude function for climate condition.

Parameter	Value	Default
{=}	{= | are | is}	–
Function	string	–

Freestream Pressure Time Function

Syntax

Freestream Pressure Time Function *{=}* *Function*

Summary

Specifies the name of freestream pressure versus time function.

Parameter	Value	Default
{=}	{= | are | is}	–
Function	string	–

Freestream Temperature

Syntax

Freestream Temperature *{=}* *Value*

Summary

Specifies a constant freestream temperature. Usage: ECKERT CONVECTIVE COEFFICIENT.

Parameter	Value	Default
{=}	{= | are | is}	–
Value	real	0.0

Freestream Temperature Altitude Function

Syntax

Freestream Temperature Altitude Function *{=}* *Function*

Summary

Specifies the freestream temperature versus altitude function for climate condition. Usage: ECKERT CONVECTIVE COEFFICIENT.

Parameter	Value	Default
{=}	{= | are | is}	–
Function	string	–

Freestream Temperature Time Function

Syntax

Freestream Temperature Time Function *{=}* *Function*

Summary

Specifies the name of freestream temperature versus time function. Usage: ECKERT CONVECTIVE COEFFICIENT.

Parameter	Value	Default
{=}	{= | are | is}	–
Function	string	–

Integrated Flux Output

Syntax

Integrated Flux Output *VariableName*

Summary

Calculate the average flux associated with this flux boundary condition.

Description

This line command specifies that, as a postprocess, the normal flux associated with this boundary condition be integrated over the surface to obtain the total power. This power is then divided by the total area of the surface to obtain the average flux on the surface, and stored in a global variable named "VariableName". This global variable may then be output to history files, or accessed in user subroutines, etc.

Parameter	Value	Default
VariableName	string	–

Integrated Power Output

Syntax

Integrated Power Output *VariableName*

Summary

Calculate the total power associated with this flux boundary condition.

Description

This line command specifies that, as a postprocess, the normal flux associated with this boundary condition be integrated over the surface to obtain the total power which is then stored into a global variable named "VariableName". This global variable may then be output to history files, or accessed in user subroutines, etc.

Parameter	Value	Default
VariableName	string	–

Mach Number

Syntax

Mach Number *{=}* *Value*

Summary

Specifies a constant Mach number. Usage: ECKERT CONVECTIVE COEFFICIENT.

Parameter	Value	Default
{=}	{= | are | is}	–
Value	real	0.0

Mach Number Time Function

Syntax

Mach Number Time Function *{=}* *Function*

Summary

Specifies the Mach number versus time function for a given flight. The tabulated time is defined relative to the START TIME. Usage: ECKERT CONVECTIVE COEFFICIENT.

Parameter	Value	Default
{=}	{= | are | is}	–
Function	string	–

Reference Density

Syntax

Reference Density *{=}* *Value*

Summary

Specifies the reference density value for the density ratio heat transfer coefficient model. Usage: DENSITY RATIO CONVECTIVE COEFFICIENT.

Parameter	Value	Default
{=}	{= | are | is}	–
Value	real	–

Reference Htc

Syntax

Reference Htc *{=}* *Value*

Summary

Specifies the reference heat transfer coefficient value for the density ratio heat transfer coefficient model. Usage: DENSITY RATIO CONVECTIVE COEFFICIENT.

Parameter	Value	Default
{=}	{= | are | is}	–
Value	real	–

Reference Temperature

Syntax

Reference Temperature *{=}* *Value*

Summary

Specify a constant reference temperature for this boundary condition.

Parameter	Value	Default
{=}	{= | are | is}	–
Value	real	–

Reference Temperature Time Function

Syntax

Reference Temperature Time Function *{=}* *FunctionName*

Summary

Specify the name of a time-dependent function for the reference temperature for this boundary condition.

Parameter	Value	Default
{=}	{= | are | is}	–
FunctionName	string	–

Start Time

Syntax

Start Time *{=}* *Value*

Summary

Specifies the analysis time at which the flight begins and the AERO heat flux will be applied.

Parameter	Value	Default
{=}	{= | are | is}	–
Value	real	0.0

Stop Time

Syntax

Stop Time *{=}* *Value*

Summary

Specifies the analysis time at which the flight ends and the AERO heat flux BC becomes inactive.

Parameter	Value	Default
{=}	{= | are | is}	–
Value	real	none

Uq Flux Multiplier

Syntax

Uq Flux Multiplier *{=}* *Value*

Summary

Specify constant scaling of the convective flux.

Description

Intended use of this scaling parameter is primarily for evaluation of model sensitivities.

Parameter	Value	Default
{=}	{= | are | is}	–
Value	real	1.0

Use Toggle Block

Syntax

Use Toggle Block *ToggleName* [*{on}* *ElementBlockList. . .*]

Summary

Specification for toggling entities in the computational model based on Toggle Block parameters. When used at the region level, the list of element blocks to be toggled must be provided. Otherwise a listing of entities is not needed as the Toggle Block will be associated with the command line or the enclosing command block.

Parameter	Value	Default
ToggleName	string	–

Convection Heat Transfer

A broad class of thermal applications involve heat transfer between a solid volume and a surrounding fluid. In numerical simulations these interactions are modeled using various surface boundary conditions depending upon the detail to which the fluid flow is resolved. If both heat transfer in the solid model and fluid flow are explicitly modeled then it is often appropriate to solve the conjugate heat transfer problem. However, in many cases the expense of this calculation may not be justified and one reverts to alternative methods of modeling these effects.

In many situations it may be appropriate to simply characterize a bulk behavior of the fluid and employ a convective flux boundary condition of the form

$$q = h(T)(T_s - T_b)$$

where \$q\$ is the heat flux per unit surface area, $h(T)$ is the surface heat transfer coefficient or convection coefficient, T_s is the surface temperature and T_b is the bulk fluid temperature. Many of the engineering models based upon experimental data for given configurations include variations of the values that enter into the expression above. In particular the values of $h(T)$ and T_b are often linked to representative fluid flow conditions.

For quick estimates of heat transfer, textbooks provide a tabulated range of surface heat transfer coefficients for general conditions. Furthermore, heat transfer handbooks provide a collection of heat transfer correlations from which convective coefficients can be computed. These correlations include empirical constants and are functions dimensionless numbers which characterize the bulk fluid flow and the bulk fluid temperature.

The Sierra Thermal module includes a number of different ways for specifying $h(T)$ and T_b. Additionally module provides a catalog of *heat transfer correlations*.

Convective Flux Boundary Condition

Scope

Aria Region, Equation System, Explicit Equation System, Root Finder Equation System

Summary

This command block specifies heat transfer on a boundary surface that can be modeled using Newton's law of cooling.

Description

Newton's law of cooling specifies that the heat flux normal to a surface is proportional to the difference between the unknown temperature of the surface and some reference temperature of the fluid in which the surface is immersed: $q_n = h(T - T_r)$. The convection coefficient, h, and the reference temperature, T_r, may be specified in several ways as explained in detail below. In particular, note that the reference temperature is usually a known quantity, because the fluid with

which it is associated is modeled as an infinite reservoir. However, if the size of this reservoir is finite, then its temperature can be affected by the energy transfer across the surface in question. This situation can be modeled by the bulk fluid element, wherein the energy of the reservoir is determined by a finite volume conservation equation.

You must specify exactly one convection coefficient, and either exactly one reference temperature or exactly one bulk fluid element name.

begin Convective Flux Boundary Condition *Name*

Add Surface *SurfaceList...*

Convective Coefficient *{=}* *Value*

Convective Coefficient Fortran Subroutine *{=}* *Name*

Convective Coefficient Node Variable *{=}* *Name*

Convective Coefficient Scale Factor *{=}* *Magnitude*

Convective Coefficient Subroutine *{=}* *Name*

Convective Coefficient Temperature Function *{=}* *NName*

Convective Coefficient Time Function *{=}* *Name*

Equation *{=}* *EquationName*

Ignore Flux Coverage

Integer Data *Values...*

Integrated Flux Output *VariableName*

Integrated Power Output *VariableName*

Real Data *Values...*

Ref Temp Convective Coefficient Subroutine *{=}* *Name* [*UsingRefTemp*]

Reference Temperature *{=}* *Value*

Reference Temperature Fortran Subroutine *{=}* *Name*

Reference Temperature Global Variable *{=}* *GlobalVariableName*

```
   Reference Temperature Node Variable {=} Name

   Reference Temperature Subroutine {=} Name

   Reference Temperature Temperature Function {=} FunctionName

   Reference Temperature Time Function {=} FunctionName

   Scaled Convective Coefficient Subroutine {=} Name FieldName {=} Field

   Scaled Ref Temp Convective Coefficient Subroutine {=} Name FieldName
↪{=} Field

   Uq Flux Multiplier {=} Value

   Use Advective Bar Name [ BulkNodes  ]

   Use Bulk Element Name

   Use Correlation Convection Model Name

   Use Data Block Name

   Use Enclosure Name

   Use Toggle Block ToggleName [ {on} ElementBlockList...  ]

   User Field Mask {=} Name [ Threshold {=} Value  ]

   User Field Scaling {=} Name

end Convective Flux Boundary Condition Name
```

Line Commands

Add Surface

Syntax

Add Surface *SurfaceList...*

Summary

Adds surfaces, by name, to a boundary condition's extent.

Description

This line command is used to add surfaces to the extent of a boundary condition. In Exodus II, surfaces are specified as side sets, that have a global integer identifier. For example, side set 12 would be added by this line command using the surface name surface_12. Note that in SIERRA, each element of an array of strings must be separated by whitespace.

Parameter	Value	Default
SurfaceList	string. . .	–

Convective Coefficient

Syntax

Convective Coefficient *{=}* *Value*

Summary

Specify a constant convective coefficient for this boundary condition.

Parameter	Value	Default
{=}	{= | are | is}	–
Value	real	–

Convective Coefficient Fortran Subroutine

Syntax

Convective Coefficient Fortran Subroutine *{=}* *Name*

Summary

Specify the name of a FORTRAN user subroutine that will be used to calculate the convective coefficient for this boundary condition.

Parameter	Value	Default
{=}	{= | are | is}	–
Name	string	–

Convective Coefficient Node Variable

Syntax

Convective Coefficient Node Variable *{=}* *Name*

Summary

Specify the name of the node variable to use for the convective coefficient that is associated with this boundary condition.

Description

The indicated node variable must be a legal Aria variable. Typically, this variable is defined by the user in the user definition command block. The node variable is interpolated to the integration points during the integration of the flux

term. Typically, this variable would be calculated in another SIERRA region, e.g., Fuego, and transferred to Aria.

Parameter	Value	Default
{=}	{= | are | is}	–
Name	string	–

Convective Coefficient Scale Factor

Syntax

Convective Coefficient Scale Factor *{=}* *Magnitude*

Summary

Specify that the convective coefficient is to be computed using sideset distribution factors that must be defined in the mesh database.

Description

The coefficient on each face is computed by interpolating the distribution factors to the Gauss points and then multiplying the result by the given magnitude. Distribution factors do not currently work with h–adaptivity, since their values are not currently interpolated to the new nodes.

Parameter	Value	Default
{=}	{= | are | is}	–
Magnitude	real	–

Convective Coefficient Subroutine

Syntax

Convective Coefficient Subroutine *{=}* *Name*

Summary

Specify the name of a user subroutine that is to be used to calculate the convective coefficient for this boundary condition.

If the user subroutine employs a user specified reference temperature model to compute the heat transfer coefficient then optional arguments USING_REF_TEMP must also appear at the end of the command line.

Parameter	Value	Default
{=}	{= | are | is}	–
Name	string	–

Convective Coefficient Temperature Function

Syntax

Convective Coefficient Temperature Function *{=}* *NName*

Summary

Specify the name of a temperature-dependent function that is to be used to calculate the convective coefficient for this boundary condition.

Parameter	Value	Default
{=}	{= | are | is}	–
NName	string	–

Convective Coefficient Time Function

Syntax

Convective Coefficient Time Function *{=}* *Name*

Summary

Specify the name of a time-dependent function that is used to calculate the convective coefficient for this boundary condition.

Parameter	Value	Default
{=}	{= | are | is}	–
Name	string	–

Equation

Syntax

Equation *{=}* *EquationName*

Summary

This command can be used to apply the Convective Flux conditions for an equation other than the ENERGY equation.

Parameter	Value	Default
{=}	{= | are | is}	–
EquationName	string	–

Ignore Flux Coverage

Syntax

Ignore Flux Coverage

Summary

This command causes the code to ignore the flux coverage Field when contact is present. Thus the flux will be applied even if the BC is being set on a contact surface.

Integer Data

Syntax

Integer Data *Values. . .*

Summary

List of integer data values to be used by the FORTRAN user subroutine. Copies of these values are provided to the subroutine hence changes to these values within the subroutine are not saved.

Parameter	Value	Default
Values	integer. . .	–

Integrated Flux Output

Syntax

Integrated Flux Output *VariableName*

Summary

Calculate the average flux associated with this flux boundary condition.

Description

This line command specifies that, as a postprocess, the normal flux associated with this boundary condition be integrated over the surface to obtain the total power. This power is then divided by the total area of the surface to obtain the average flux on the surface, and stored in a global variable named "VariableName". This global variable may then be output to history files, or accessed in user subroutines, etc.

Parameter	Value	Default
VariableName	string	–

Integrated Power Output

Syntax

Integrated Power Output *VariableName*

Summary

Calculate the total power associated with this flux boundary condition.

Description

This line command specifies that, as a postprocess, the normal flux associated with this boundary condition be integrated over the surface to obtain the total power which is then stored into a global variable named "VariableName". This global variable may then be output to history files, or accessed in user subroutines, etc.

Parameter	Value	Default
VariableName	string	–

Real Data

Syntax

Real Data *Values. . .*

Summary

List of real data values to be used by the FORTRAN user subroutine. Copies of these values are provided to the subroutine hence changes to these values within the subroutine are not saved.

Parameter	Value	Default
Values	real. . .	–

Ref Temp Convective Coefficient Subroutine

Syntax

Ref Temp Convective Coefficient Subroutine *{=}* *Name* [*UsingRefTemp*]

Summary

Specify the name of a user subroutine that is to be used to calculate the convective coefficient for this boundary condition. Use of this command line will enable the reference temperature to be supplied to the subroutine interface.

Parameter	Value	Default
{=}	{= | are | is}	–
Name	string	–

Reference Temperature

Syntax

Reference Temperature *{=}* *Value*

Summary

Specify a constant reference temperature for this boundary condition.

Parameter	Value	Default
{=}	{= | are | is}	–
Value	real	–

Reference Temperature Fortran Subroutine

Syntax

Reference Temperature Fortran Subroutine *{=}* *Name*

Summary

Specifies the name of a FORTRAN user-defined subroutine that will be used to calculate the reference temperature associated with this boundary condition.

Parameter	Value	Default
{=}	{= | are | is}	–
Name	string	–

Reference Temperature Global Variable

Syntax

Reference Temperature Global Variable *{=}* *GlobalVariableName*

Summary

Specify a global variable to be used for reference temperature.

Parameter	Value	Default
{=}	{= | are | is}	–
GlobalVariableName	string	–

Reference Temperature Node Variable

Syntax

Reference Temperature Node Variable *{=}* *Name*

Summary

Specify the name of the node variable to use for the reference temperature that is associated with this boundary condition.

Description

The indicated node variable must be a legal Aria variable. Typically, this variable is defined by the user in the user definition command block. The node variable is interpolated to the integration points during the integration of the flux term. Typically, this variable would be calculated in another SIERRA region, e.g., Fuego, and transferred to Aria.

Parameter	Value	Default
{=}	{= | are | is}	–
Name	string	–

Reference Temperature Subroutine

Syntax

Reference Temperature Subroutine *{=}* *Name*

Summary

Specifies the name of a user-defined subroutine that is to be used to calculate the reference temperature associated with this boundary condition.

Parameter	Value	Default
{=}	{= | are | is}	–
Name	string	–

Reference Temperature Temperature Function

Syntax

Reference Temperature Temperature Function *{=}* *FunctionName*

Summary

Specifies the name of the temperature-dependent function that is to be used to calculate the reference temperature associated with this boundary condition.

Parameter	Value	Default
{=}	{= | are | is}	–
FunctionName	string	–

Reference Temperature Time Function

Syntax

Reference Temperature Time Function *{=}* *FunctionName*

Summary

Specify the name of a time-dependent function for the reference temperature for this boundary condition.

Parameter	Value	Default
{=}	{= | are | is}	–
FunctionName	string	–

Scaled Convective Coefficient Subroutine

Syntax

Scaled Convective Coefficient Subroutine *{=}* *Name* *FieldName* *{=}* *Field*

Summary

Specify the name of a user subroutine that is to be used to calculate the scaled convective coefficient for this boundary condition. The interpolated scaling Field is supplied directly to the user subroutine.

Parameter	Value	Default
{=}	{= | are | is}	–
Name	string	–
FieldName	string	–
{=}	{= | are | is}	–
Field	string	–

Scaled Ref Temp Convective Coefficient Subroutine

Syntax

Scaled Ref Temp Convective Coefficient Subroutine *{=}* *Name* *FieldName* *{=}* *Field*

Summary

Specify the name of a user subroutine that is to be used to calculate the scaled convective coefficient for this boundary condition. Use of this command will enable the reference temperature to be supplied to the user subroutine. Additionally interpolated values of the scaling Field will be delivered to the subroutine.

Parameter	Value	Default
{=}	{= | are | is}	–
Name	string	–
FieldName	string	–
{=}	{= | are | is}	–
Field	string	–

Uq Flux Multiplier

Syntax

Uq Flux Multiplier *{=}* *Value*

Summary

Specify constant scaling of the convective flux.

Description

Intended use of this scaling parameter is primarily for evaluation of model sensitivities.

Parameter	Value	Default
{=}	{= | are | is}	–
Value	real	1.0

Use Advective Bar

Syntax

Use Advective Bar *Name* [*BulkNodes*]

Summary

Use the named advective bar to model the reference temperature.

Description

This line command specifies the name of an advective bar that has been defined using the advective bar command block. The temperature of the advective bar is used as the reference temperature, T_r, for the convective heat transfer and this is

computed based on a geometric coupling algorithm. Note that it is illegal to specify both a reference temperature and an advective bar.

Parameter	Value	Default
Name	string	–

Use Bulk Element

Syntax

Use Bulk Element *Name*

Summary

Use the named bulk element to model the reference temperature.

Description

This line command specifies the name of a bulk fluid element that has been defined using the bulk fluid command block. The temperature of the bulk fluid element is used as the reference temperature, T_r, for the convective heat transfer.

Note: It is illegal to specify both a reference temperature and a bulk element.

Parameter	Value	Default
Name	string	–

Use Correlation Convection Model

Syntax

Use Correlation Convection Model *Name*

Summary

Specifies correlation model for convection coefficient

Parameter	Value	Default
Name	string	–

Use Data Block

Syntax

Use Data Block *Name*

Summary

Reference to predefined data to be used by the user subroutine. These values may be changed by the user subroutine.

Parameter	Value	Default
Name	string	–

Use Enclosure

Syntax

Use Enclosure *Name*

Summary

Use the named enclosure to define the surface list and possibly model the reference temperature using the MBL bulk node of the enclosure.

Description

This line command specifies the name of an enclosure that has been defined using the enclosure definition command block. The list of surfaces for this convective BC is taken from the enclosure surfaces and it is illegal to specify both an enclosure and a surface list. If the enclosure has the Mean Beam Length (MBL) model activated, then this convective BC is linked to the bulk fluid element associated with the MBL model. The temperature of the bulk fluid element is used as the reference temperature, T_r, for the convective heat transfer.

Note: It is illegal to specify both a reference temperature and an enclosure with an MBL bulk element.

Parameter	Value	Default
Name	string	–

Use Toggle Block

Syntax

Use Toggle Block *ToggleName* [*{on} ElementBlockList. . .*]

Summary

Specification for toggling entities in the computational model based on Toggle Block parameters. When used at the region level, the list of element blocks to be toggled must be provided. Otherwise a listing of entities is not needed as the Toggle Block will be associated with the command line or the enclosing command block.

Parameter	Value	Default
ToggleName	string	–

User Field Mask

Syntax

User Field Mask *{=}* *Name* [*Threshold* *{=}* *Value*]

Summary

Specify that the convective boundary condition is to be masked by nonzero values of a user defined Field.

Description

The convective boundary condition is applied in a conventional manner but then masked by nonzero values of an interpolated user defined Field. In most cases the user Field is transferred to Aria but it could also be computed. Default threshold for the masked value is 0.0, i.e. mask value equals 1.0 for values greater than zero.

Parameter	Value	Default
{=}	{= | are | is}	–
Name	string	–

User Field Scaling

Syntax

User Field Scaling *{=}* *Name*

Summary

Specify that the convective boundary condition is to be scaled by a user defined Field.

Description

The convective boundary condition is applied in a conventional manner but then scaled by an interpolated user defined Field. In most cases the user Field is transferred to Aria and acts as a weighting of the flux condition.

Parameter	Value	Default
{=}	{= | are | is}	–
Name	string	–

Heat Flux Boundary Condition

Scope

Aria Region, Equation System, Explicit Equation System, Root Finder Equation System

Summary

This command block specifies that a known heat flux is to be applied normal to

the given surface.

Description

The flux can vary in arbitrary way, e.g. it may be a specified constant, a function of temperature, time etc. You may only specify one kind of flux e.g it is not legal to specify a constant value and a time dependent function. If you want to add two such fluxes on a single surface, then either write a user subroutine or use two different command blocks. If a function name is given then the function must be defined in a function definition command block.

begin Heat Flux Boundary Condition *Name*

Add Surface *SurfaceList...*

Element Subroutine *{=} Name*

Equation *{=} EquationName*

Field Scaling *{=} Scale_factor*

Flux *{=} Value*

Flux Fortran Subroutine *{=} Name*

Flux Node Variable *{=} Name* [*Multiplier {=} Value*]

Flux Scale Factor *{=} Value*

Flux Temperature Function *{=} Namep*

Flux Time Function *{=} Name*

Flux Vector Node Variable *{=} Name* [*Multiplier {=} Value*]

Ignore Flux Coverage

Influx Vector Node Variable *{=} Name* [*Multiplier {=} Value*]

Integer Data *Values...*

Integrated Flux Output *VariableName*

Integrated Power Output *VariableName*

Real Data *Values...*

```
   Scaling Time Function {=} Function

   Scaling With Global Variable {=} name

   Use Data Block Name

   Use Toggle Block ToggleName [ {on} ElementBlockList...  ]

end Heat Flux Boundary Condition Name
```

Line Commands

Add Surface

Syntax

Add Surface *SurfaceList...*

Summary

Adds surfaces, by name, to a boundary condition's extent.

Description

This line command is used to add surfaces to the extent of a boundary condition. In Exodus II, surfaces are specified as side sets, that have a global integer identifier. For example, side set 12 would be added by this line command using the surface name surface_12. Note that in SIERRA, each element of an array of strings must be separated by whitespace.

Parameter	Value	Default
SurfaceList	string...	–

Element Subroutine

Syntax

Element Subroutine *{=} Name*

Summary

Specifies that the named user-defined subroutine be used. At most one subroutine name may be specified for a given quantity. This subroutine must conform to the "element signature" argument list type.

Parameter	Value	Default
{=}	{= | are | is}	–
Name	string	–

Equation

Syntax

Equation *{=}* *EquationName*

Summary

This command can be used to apply the Heat Flux conditions for an equation other than the ENERGY equation.

Parameter	Value	Default
{=}	{= | are | is}	–
EquationName	string	–

Field Scaling

Syntax

Field Scaling *{=}* *Scale_factor*

Summary

This command is only relevant when the FLUX NODE VARIABLE, FLUX VECTOR NODE VARIABLE or INFLUX VECTOR NODE VARIABLE command line option is present. The command enables scaling of the Field by a single value for the surfaces defined within the scope of the current Heat Flux command block. For vector Fields all components will be scaled by the provided value.

Parameter	Value	Default
{=}	{= | are | is}	–
Scale_factor	real	–

Flux

Syntax

Flux *{=}* *Value*

Summary

This line command specifies a constant value of heat flux.

Parameter	Value	Default
{=}	{= | are | is}	–
Value	real	–

Flux Fortran Subroutine

Syntax

Flux Fortran Subroutine *{=}* *Name*

Summary

Specify the name of a FORTRAN user subroutine that will be used to calculate the flux boundary condition.

Parameter	Value	Default
{=}	{= | are | is}	–
Name	string	–

Flux Node Variable

Syntax

Flux Node Variable *{=}* *Name* [*Multiplier* *{=}* *Value*]

Summary

Specify the name of the node variable to use for the flux that is associated with this boundary condition. Here the node variable is assumed to be the product of the surface unit normal vector with the nodal heat flux. Optionally, the value can be scaled by product of Multiplier value and FIELD SCALING.

Description

The indicated node variable must be a legal Aria variable. In most cases this variable is defined by the user in the user variable definition command block. The node variable is interpolated to the quadrature points during the evaluation of flux term contributions. If need be, both the sign and magnitude of the node variable can be modified by using the FIELD SCALING command line. Typically, the node variable would be calculated in another SIERRA region, e.g., Fuego, and transferred to Aria.

Parameter	Value	Default
{=}	{= | are | is}	–
Name	string	–

Flux Scale Factor

Syntax

Flux Scale Factor *{=}* *Value*

Summary

Specifies the magnitude by which the distribution factors will be multiplied in order to determine the boundary condition. If this line command is used, sideset distribution factors must be defined in the input mesh for all surfaces which are associated with this boundary condition.

Description

The flux on each face is computed by interpolating the distribution factors to the Gauss points and then multiplying the result by the given magnitude.

Distribution factors do not currently work with h–adaptivity, since their values are not currently interpolated to the new nodes.

Parameter	Value	Default
{=}	{= | are | is}	–
Value	real	–

Flux Temperature Function

Syntax

Flux Temperature Function *{=}* *Namep*

Summary

Specifies the name of a temperature dependent heat flux function that is used to calculate the normal flux.

Parameter	Value	Default
{=}	{= | are | is}	–
Namep	string	–

Flux Time Function

Syntax

Flux Time Function *{=}* *Name*

Summary

Specifies the name of a time dependent heat flux function that is used to calculate the normal flux.

Parameter	Value	Default
{=}	{= | are | is}	–
Name	string	–

Flux Vector Node Variable

Syntax

Flux Vector Node Variable *{=}* *Name* [*Multiplier* *{=}* *Value*]

Summary

Specify the name of the vector node variable to use for the flux that is associated with this boundary condition. Here the flux contribution for a surface will be evaluated by forming a dot product of the unit normal vector with the vector node variable, without regard to the sign of the dot product. Optionally, the vector values can be scaled by product of Multiplier value and FIELD SCALING.

Description

The indicated node variable must be a legal Aria vector variable. In most cases

this variable is defined by the user in the user variable definition command block. The node variable is interpolated to the quadrature points during the evaluation of flux term contributions. If need be, both the sign and magnitude of the node variable can be modified by using the FIELD SCALING command line. Typically, the vector node variable would be calculated in another SIERRA region, e.g., Fuego, and transferred to Aria.

Parameter	Value	Default
{=}	{= | are | is}	–
Name	string	–

Ignore Flux Coverage

Syntax

Ignore Flux Coverage

Summary

This command causes the code to ignore the flux coverage Field when contact is present. Thus the flux will be applied even if the BC is being set on a contact surface.

Influx Vector Node Variable

Syntax

Influx Vector Node Variable *{=}* *Name* [*Multiplier* *{=}* *Value*]

Summary

Specify the name of the node variable to use for the flux that is associated with this boundary condition. Here the applied flux will be formed by the dot product of the surface unit normal vector with the nodal variable which produce an influx contribution. Optionally, the value can be scaled by product of Multiplier value and FIELD SCALING.

Description

The indicated node variable must be a legal Aria variable. In most cases this variable is defined by the user in the User Variable definition command block. The node variable is interpolated to the quadrature points during the evaluation of flux term contributions. If need be, both the sign and magnitude of the node variable can be modified by using the FIELD SCALING command line. Typically, the node variable would be calculated in another SIERRA region, e.g., Fuego, and transferred to Aria.

Parameter	Value	Default
{=}	{= | are | is}	–
Name	string	–

Integer Data

Syntax

Integer Data *Values. . .*

Summary

List of integer data values to be used by the FORTRAN user subroutine. Copies of these values are provided to the subroutine hence changes to these values within the subroutine are not saved.

Parameter	Value	Default
Values	integer. . .	–

Integrated Flux Output

Syntax

Integrated Flux Output *VariableName*

Summary

Calculate the average flux associated with this flux boundary condition.

Description

This line command specifies that, as a postprocess, the normal flux associated with this boundary condition be integrated over the surface to obtain the total power. This power is then divided by the total area of the surface to obtain the average flux on the surface, and stored in a global variable named "VariableName". This global variable may then be output to history files, or accessed in user subroutines, etc.

Parameter	Value	Default
VariableName	string	–

Integrated Power Output

Syntax

Integrated Power Output *VariableName*

Summary

Calculate the total power associated with this flux boundary condition.

Description

This line command specifies that, as a postprocess, the normal flux associated with this boundary condition be integrated over the surface to obtain the total power which is then stored into a global variable named "VariableName". This global variable may then be output to history files, or accessed in user subroutines, etc.

Parameter	Value	Default
VariableName	string	–

Real Data

Syntax

Real Data *Values. . .*

Summary

List of real data values to be used by the FORTRAN user subroutine. Copies of these values are provided to the subroutine hence changes to these values within the subroutine are not saved.

Parameter	Value	Default
Values	real. . .	–

Scaling Time Function

Syntax

Scaling Time Function *{=}* *Function*

Summary

This command is only relevant when the FLUX NODE VARIABLE, FLUX VECTOR NODE VARIABLE or INFLUX VECTOR NODE VARIABLE command line option is present. This command enables time function scaling of the field variable for the surfaces defined within the scope of the current Heat Flux command block.

Parameter	Value	Default
{=}	{= | are | is}	–
Function	string	–

Scaling With Global Variable

Syntax

Scaling With Global Variable *{=}* *name*

Summary

This command is only relevant when the FLUX NODE VARIABLE, FLUX VECTOR NODE VARIABLE or INFLUX VECTOR NODE VARIABLE command line option is present. The command enables scaling of the Field by a global variable for the surfaces defined within the Heat Flux command block. For vector Fields all components will be scaled by the named scalar global variable. The scalar global variable itself must be defined at the Region scope as type Real with length 1.

Parameter	Value	Default
{=}	{= | are | is}	–
name	string	–

Use Data Block

Syntax

Use Data Block *Name*

Summary

Reference to predefined data to be used by the user subroutine. These values may be changed by the user subroutine.

Parameter	Value	Default
Name	string	–

Use Toggle Block

Syntax

Use Toggle Block *ToggleName* [*{on} ElementBlockList. . .*]

Summary

Specification for toggling entities in the computational model based on Toggle Block parameters. When used at the region level, the list of element blocks to be toggled must be provided. Otherwise a listing of entities is not needed as the Toggle Block will be associated with the command line or the enclosing command block.

Parameter	Value	Default
ToggleName	string	–

Radiative Flux Boundary Condition

Scope

Aria Region, Equation System, Explicit Equation System, Root Finder Equation System

Summary

Radiative heat flux boundary condition. You must specify either a reference temperature or provide a nodal field for irradiation.

Description

This boundary condition models the radiative heat transfer from a specified surface to a surrounding surface. The temperature of the surrounding surface is assumed to have enough thermal mass that its temperature is unaffected by changes in computed temperature, and is therefore known. The heat flux normal to the surface is is given by $q_n = \varepsilon F(\sigma T^4 - G)$, where the incident radiative flux, or irradiation, G, may either be specified directly as a nodal field, or may be specified through a reference temperature, $G = \sigma T_r^4$. The Stefan-Boltzmann

constant, σ, is defined in a Global Constants command block, ε is the surface emissivity, and F is the form factor.

With the exception of σ, each of these parameters may vary in one of several ways, as defined below. The emissivity, if left unspecified, is determined locally from the material associated with the element that underlies each face on the surface. In the case of faces that are created as a result of element death, the emissivity is always determined by the underlying element.

begin Radiative Flux Boundary Condition *Name*

Add Surface *SurfaceList...*

Emissivity *{=}* *Value* [On *SurfaceName*]

Emissivity For Bands *{=}* *Emissivity...*

Emissivity Fortran Subroutine *{=}* *Name*

Emissivity Function *{=}* *FunctionName* [On *SurfaceName*]

Emissivity Subroutine *{=}* *MySub* [On *SurfaceName*]

Emissivity Time Function *{=}* *FunctionName* [On *SurfaceName*]

Equation *{=}* *EquationName*

Form Factor Fortran Subroutine *{=}* *Name*

Ignore Flux Coverage

Integer Data *Values...*

Integrated Flux Output *VariableName*

Integrated Power Output *VariableName*

Irradiation Node Variable *{=}* *Name*

Irradiation Subroutine *{=}* *Name*

Irradiation Time Function *{=}* *Name*

Radiation Form Factor *{=}* *Value*

Radiation Form Factor Subroutine *{=}* *Name*

```
   Radiation Form Factor Temperature Function {=} Name

   Radiation Form Factor Time Function {=} Name

   Real Data Values...

   Reference Temperature {=} Value

   Reference Temperature Fortran Subroutine {=} Name

   Reference Temperature Global Variable {=} GlobalVariableName

   Reference Temperature Subroutine {=} Name

   Reference Temperature Temperature Function {=} FunctionName

   Reference Temperature Time Function {=} FunctionName

   Use Advective Bar Name [ BulkNodes  ]

   Use Banded Wavelength Model ModelName

   Use Bulk Element Name

   Use Data Block Name

   Use Toggle Block ToggleName [ {on} ElementBlockList...  ]

end Radiative Flux Boundary Condition Name
```

Line Commands

Add Surface

Syntax

Add Surface *SurfaceList...*

Summary

Adds surfaces, by name, to a boundary condition's extent.

Description

This line command is used to add surfaces to the extent of a boundary condition. In Exodus II, surfaces are specified as side sets, that have a global integer identifier. For example, side set 12 would be added by this line command using

the surface name surface_12. Note that in SIERRA, each element of an array of strings must be separated by whitespace.

Parameter	Value	Default
SurfaceList	string. . .	–

Emissivity

Syntax

Emissivity *{=}* *Value* [On *SurfaceName*]

Summary

Sets a constant value of emissivity for a defined surface. If the optional parameters are not included, then this is the default emissivity for the enclosure. Otherwise it is only applied to the indicated surface.

Note: If a surface is called out more than once, the emissivity definition is overwritten: last one in wins.

Parameter	Value	Default
{=}	{= | are | is}	–
Value	real	–

Emissivity For Bands

Syntax

Emissivity For Bands *{=}* *Emissivity. . .*

Summary

Specify the constant emissivities for each wavelength band.

Parameter	Value	Default
{=}	{= | are | is}	–
Emissivity	real. . .	–

Emissivity Fortran Subroutine

Syntax

Emissivity Fortran Subroutine *{=}* *Name*

Summary

Specify the name of a FORTRAN user subroutine that will be used to calculate the convective coefficient for this boundary condition.

Parameter	Value	Default
{=}	{= | are | is}	–
Name	string	–

Emissivity Function

Syntax

Emissivity Function *{=}* *FunctionName* [On *SurfaceName*]

Summary

Sets a emissivity function for a defined surface. If the optional parameters are not included, then this is the default emissivity for the enclosure. Otherwise it is only applied to the indicated surface.

Parameter	Value	Default
{=}	{= | are | is}	–
FunctionName	string	–

Emissivity Subroutine

Syntax

Emissivity Subroutine *{=}* *MySub* [On *SurfaceName*]

Summary

Sets a emissivity user subroutine for a defined surface. If the optional parameters are not included, then this is the default emissivity for the enclosure. Otherwise it is only applied to the indicated surface.

Also, the software supports using locally scoped user data for most user subroutines, but I haven't figured out a syntax for it here yet. So it is not yet supported. If you need to get data into this subroutine, use the region's "REAL DATA" and "INTEGER DATA" line commands.

Note: If a surface is called out more than once, the emissivity definition is overwritten: last one in wins.

Parameter	Value	Default
{=}	{= | are | is}	–
MySub	string	–

Emissivity Time Function

Syntax

Emissivity Time Function *{=}* *FunctionName* [On *SurfaceName*]

Summary

Sets a emissivity function of time for a defined surface. If the optional parameters are not included, then this is the default emissivity for the enclosure. Otherwise it is only applied to the indicated surface.

Parameter	Value	Default
{=}	{= | are | is}	–
FunctionName	string	–

Equation

Syntax

Equation *{=}* *EquationName*

Summary

This command can be used to apply the Radiative Flux conditions for an equation other than the ENERGY equation.

Parameter	Value	Default
{=}	{= | are | is}	–
EquationName	string	–

Form Factor Fortran Subroutine

Syntax

Form Factor Fortran Subroutine *{=}* *Name*

Summary

Specify the name of a FORTRAN user subroutine that will be used to calculate the form factor for this boundary condition.

Parameter	Value	Default
{=}	{= | are | is}	–
Name	string	–

Ignore Flux Coverage

Syntax

Ignore Flux Coverage

Summary

This command causes the code to ignore the flux coverage Field when contact is present. Thus the flux will be applied even if the BC is being set on a contact surface.

Integer Data

Syntax

Integer Data *Values. . .*

Summary

List of integer data values to be used by the FORTRAN user subroutine. Copies of these values are provided to the subroutine hence changes to these values within the subroutine are not saved.

Parameter	Value	Default
Values	integer. . .	–

Integrated Flux Output

Syntax

Integrated Flux Output *VariableName*

Summary

Calculate the average flux associated with this flux boundary condition.

Description

This line command specifies that, as a postprocess, the normal flux associated with this boundary condition be integrated over the surface to obtain the total power. This power is then divided by the total area of the surface to obtain the average flux on the surface, and stored in a global variable named "VariableName". This global variable may then be output to history files, or accessed in user subroutines, etc.

Parameter	Value	Default
VariableName	string	–

Integrated Power Output

Syntax

Integrated Power Output *VariableName*

Summary

Calculate the total power associated with this flux boundary condition.

Description

This line command specifies that, as a postprocess, the normal flux associated with this boundary condition be integrated over the surface to obtain the total power which is then stored into a global variable named "VariableName". This global variable may then be output to history files, or accessed in user subroutines, etc.

Parameter	Value	Default
VariableName	string	–

Irradiation Node Variable

Syntax

Irradiation Node Variable *{=}* *Name*

Summary

Specifies the name of the node variable in which the incident radiative flux associated with this boundary condition is stored.

Description

The indicated node variable must be a valid user variable. Typically, this variable is defined by the user within the Region scope. The node variable is interpolated to the integration points during the integration of the flux term. Typically, this variable would be calculated in another SIERRA region, e.g., Fuego, and populated by transfer in the appropriate Region.

Parameter	Value	Default
{=}	{= | are | is}	–
Name	string	–

Irradiation Subroutine

Syntax

Irradiation Subroutine *{=}* *Name*

Summary

Specify the name of a user subroutine used to evaluate the irradiation for this boundary condition.

Parameter	Value	Default
{=}	{= | are | is}	–
Name	string	–

Irradiation Time Function

Syntax

Irradiation Time Function *{=}* *Name*

Summary

Specifies the name of a time-dependent function that is used to calculate the irradiation associated with this boundary condition.

Parameter	Value	Default
{=}	{= | are | is}	–
Name	string	–

Radiation Form Factor

Syntax

Radiation Form Factor *{=}* *Value*

Summary

Specifies a constant value for the radiation form factor for this boundary condition.

Description

For two surfaces, the BC surface (1) and far-field surface (2), ε denoting emissivity, A denoting area, and F_{12} is the view factor.

$$F = \begin{cases} 1 & \text{if } A_2 \gg A_1 \\ F_{12}/[(1-\varepsilon_1)F_{12}+\varepsilon_1] & \text{for } \varepsilon_1 = 1 \text{ and } A_2 \text{ not } \gg A_1 \\ F_{12} & \text{if } \varepsilon_1 = \varepsilon_2 = 1 \end{cases}$$

Parameter	Value	Default
{=}	{= | are | is}	–
Value	real	–

Radiation Form Factor Subroutine

Syntax

Radiation Form Factor Subroutine *{=}* *Name*

Summary

Specifies the name of a user subroutine that is used to calculate the radiation form factor associated with this boundary condition.

Parameter	Value	Default
{=}	{= | are | is}	–
Name	string	–

Radiation Form Factor Temperature Function

Syntax

Radiation Form Factor Temperature Function *{=}* *Name*

Summary

Specifies the name of a temperature-dependent function that is used to calculate the radiation form factor associated with this boundary condition.

Parameter	Value	Default
{=}	{= | are | is}	–
Name	string	–

Radiation Form Factor Time Function

Syntax

Radiation Form Factor Time Function *{=}* *Name*

Summary

Specifies the name of a time-dependent function that is used to calculate the form factor associated with this boundary condition.

Parameter	Value	Default
{=}	{= | are | is}	–
Name	string	–

Real Data

Syntax

Real Data *Values. . .*

Summary

List of real data values to be used by the FORTRAN user subroutine. Copies of these values are provided to the subroutine hence changes to these values within the subroutine are not saved.

Parameter	Value	Default
Values	real. . .	–

Reference Temperature

Syntax

Reference Temperature *{=}* *Value*

Summary

Specify a constant reference temperature for this boundary condition.

Parameter	Value	Default
{=}	{= | are | is}	–
Value	real	–

Reference Temperature Fortran Subroutine

Syntax

Reference Temperature Fortran Subroutine *{=}* *Name*

Summary

Specifies the name of a FORTRAN user-defined subroutine that will be used to calculate the reference temperature associated with this boundary condition.

Parameter	Value	Default
{=}	{= | are | is}	–
Name	string	–

Reference Temperature Global Variable

Syntax

Reference Temperature Global Variable *{=} GlobalVariableName*

Summary

Specify a global variable to be used for reference temperature.

Parameter	Value	Default
{=}	{= | are | is}	–
GlobalVariableName	string	–

Reference Temperature Subroutine

Syntax

Reference Temperature Subroutine *{=} Name*

Summary

Specifies the name of a user-defined subroutine that is to be used to calculate the reference temperature associated with this boundary condition.

Parameter	Value	Default
{=}	{= | are | is}	–
Name	string	–

Reference Temperature Temperature Function

Syntax

Reference Temperature Temperature Function *{=} FunctionName*

Summary

Specifies the name of the temperature-dependent function that is to be used to calculate the reference temperature associated with this boundary condition.

Parameter	Value	Default
{=}	{= | are | is}	–
FunctionName	string	–

Reference Temperature Time Function

Syntax

Reference Temperature Time Function *{=} FunctionName*

Summary

Specify the name of a time-dependent function for the reference temperature for this boundary condition.

Parameter	Value	Default
{=}	{= | are | is}	–
FunctionName	string	–

Use Advective Bar

Syntax

Use Advective Bar *Name* [*BulkNodes*]

Summary

Use the named advective bar to model the reference temperature.

Description

This line command specifies the name of an advective bar that has been defined using the advective bar command block. The temperature of the advective bar is used as the reference temperature, T_r, for the convective heat transfer and this is computed based on a geometric coupling algorithm. Note that it is illegal to specify both a reference temperature and an advective bar.

Parameter	Value	Default
Name	string	–

Use Banded Wavelength Model

Syntax

Use Banded Wavelength Model *ModelName*

Summary

Use a specified banded wavelength model for use in enclosure radiation.

Description

Requests that a specific banded wavelength model named ModelName, be associated with an enclosure, where the ModelName is defined outside of the enclosure definition block. Only one BANDED WAVELENGTH MODEL can be used for an enclosure.

Parameter	Value	Default
ModelName	string	–

Use Bulk Element

Syntax

Use Bulk Element *Name*

Summary

Use the named bulk element to model the reference temperature.

Description

This line command specifies the name of a bulk fluid element that has been defined using the bulk fluid command block. The temperature of the bulk fluid element is used as the reference temperature, T_r, for the convective heat transfer.

Note: It is illegal to specify both a reference temperature and a bulk element.

Parameter	Value	Default
Name	string	–

Use Data Block

Syntax

Use Data Block *Name*

Summary

Reference to predefined data to be used by the user subroutine. These values may be changed by the user subroutine.

Parameter	Value	Default
Name	string	–

Use Toggle Block

Syntax

Use Toggle Block *ToggleName* [*{on} ElementBlockList. . .*]

Summary

Specification for toggling entities in the computational model based on Toggle Block parameters. When used at the region level, the list of element blocks to be toggled must be provided. Otherwise a listing of entities is not needed as the Toggle Block will be associated with the command line or the enclosing command block.

Parameter	Value	Default
ToggleName	string	–

Temperature Boundary Condition

Scope

Aria Region, Equation System, Explicit Equation System, Root Finder Equation System

Summary

Specified temperature boundary condition.

Description

For this boundary condition, the conservation equations are discarded, and the temperature is completely determined by this boundary condition. The temperature specification must be of exactly one type: e.g., constant, time-dependent, or defined in a user subroutine. If a user subroutine is used, real and integer data may be declared that will be local in scope to this instance of the boundary condition. This data may be accessed using a user query function.

```
begin Temperature Boundary Condition BC name

   Add Surface SurfaceList...

   Field {=} VariableName

   Integer Data Values...

   Node Subroutine {=} Name

   Real Data Values...

   Temperature {=} Value

   Temperature Fortran Subroutine {=} Name

   Temperature Node Variable {=} Name

   Temperature Scale Factor {=} Magnitude

   Temperature Time Function {=} FunctionName

   Use Data Block Name

   Use Death Name

   Use Toggle Block ToggleName [ {on} ElementBlockList...  ]

   Value {=} VariableValue
```

end Temperature Boundary Condition *BC name*

Line Commands

Add Surface

Syntax

Add Surface *SurfaceList. . .*

Summary

Adds surfaces, by name, to a boundary condition's extent.

Description

This line command is used to add surfaces to the extent of a boundary condition. In Exodus II, surfaces are specified as side sets, that have a global integer identifier. For example, side set 12 would be added by this line command using the surface name surface_12. Note that in SIERRA, each element of an array of strings must be separated by whitespace.

Parameter	Value	Default
SurfaceList	string. . .	–

Field

Syntax

Field *{=}* *VariableName*

Summary

This command can be used with Dirichlet conditions for any solution unknown used in Aria. The FIELD command argument defines the name of the variable to be specified with this boundary condition.

Parameter	Value	Default
{=}	{= | are | is}	–
VariableName	string	–

Integer Data

Syntax

Integer Data *Values. . .*

Summary

List of integer data values to be used by the FORTRAN user subroutine. Copies of these values are provided to the subroutine hence changes to these values within the subroutine are not saved.

Parameter	Value	Default
Values	integer. . .	–

Node Subroutine

Syntax

Node Subroutine *{=}* *Name*

Summary

Specifies that the named user-defined subroutine be used. At most one subroutine name may be specified for a given quantity. This subroutine must conform to the “node signature” argument list type.

Parameter	Value	Default
{=}	{= | are | is}	–
Name	string	–

Real Data

Syntax

Real Data *Values. . .*

Summary

List of real data values to be used by the FORTRAN user subroutine. Copies of these values are provided to the subroutine hence changes to these values within the subroutine are not saved.

Parameter	Value	Default
Values	real. . .	–

Temperature

Syntax

Temperature *{=}* *Value*

Summary

Specify the constant value of the temperature. Using this command is equivalent to using both the FIELD = temperature and VALUE = real_value command lines.

Parameter	Value	Default
{=}	{= | are | is}	–
Value	real	–

Temperature Fortran Subroutine

Syntax

Temperature Fortran Subroutine *{=}* *Name*

Summary

Specify the name of a FORTRAN user subroutine that will be used to calculate the nodal temperature boundary condition.

Parameter	Value	Default
{=}	{= | are | is}	–
Name	string	–

Temperature Node Variable

Syntax

Temperature Node Variable *{=}* *Name*

Summary

Specify the name of the node variable to use for the temperature associated with this boundary condition.

Description

The indicated node variable must be a legal Aria variable. This variable is defined from the input file in the USER VARIABLE command block or with the USER FIELD command. Two distinct use cases may arise, 1) the variable is calculated in another SIERRA region, e.g., Fuego, and transferred to Aria 2) the variable exists in the input grid file and is populated using the READ VARIABLE command from within the USER FIELD command block.

Parameter	Value	Default
{=}	{= | are | is}	–
Name	string	–

Temperature Scale Factor

Syntax

Temperature Scale Factor *{=}* *Magnitude*

Summary

Specifies the magnitude by which the distribution factors will be multiplied in order to determine the boundary condition. If this line command is used, node distribution factors must be defined in the input mesh for all nodesets which are associated with this boundary condition.

Description

If this line command is used, node distribution factors must be defined in the input mesh for all nodesets which are associated with this boundary condition. There is currently an inconsistency with respect to temperature boundary

conditions and distribution factors: It is not legal to specify sideset distribution factors with the temperature boundary condition: nodesets must be used. Distribution factors also do not work with h-adaptivity, as their values are not currently interpolated to the new nodes. In any case, for temperature boundary conditions, in order for this interpolation to occur, then a sideset would have to be used, which is not supported at this time.

Parameter	Value	Default
{=}	{= | are | is}	–
Magnitude	real	–

Temperature Time Function

Syntax

Temperature Time Function *{=}* *FunctionName*

Summary

Name of the time-dependent function that specifies the temperature.

Parameter	Value	Default
{=}	{= | are | is}	–
FunctionName	string	–

Use Data Block

Syntax

Use Data Block *Name*

Summary

Reference to predefined data to be used by the user subroutine. These values may be changed by the user subroutine.

Parameter	Value	Default
Name	string	–

Use Death

Syntax

Use Death *Name*

Summary

Associates this boundary condition with element faces that are created as a result of element death due to the specified element death command block. Note that this line command makes the owning boundary condition have a dynamic extent that changes as elements die.

Description

This line command specifies that the faces that are created as a result of the element death criterion in the named element death command block are added to the extent of this boundary condition. A boundary condition may specify an extent via the "add surface" line command, or the "use death" line command. In this way it is possible to have a boundary condition that is initially empty and then grows dynamically with the surface created as a result of the element death. At least one of these methods must be used.

A boundary condition may be associated with more than one death command block by including this line command more than once. However, to avoid issues of precedence, there is a rule that a given element may not die for more than one reason.

Parameter	Value	Default
Name	string	–

Use Toggle Block

Syntax

Use Toggle Block *ToggleName* [*{on}* *ElementBlockList. . .*]

Summary

Specification for toggling entities in the computational model based on Toggle Block parameters. When used at the region level, the list of element blocks to be toggled must be provided. Otherwise a listing of entities is not needed as the Toggle Block will be associated with the command line or the enclosing command block.

Parameter	Value	Default
ToggleName	string	–

Value

Syntax

Value *{=}* *VariableValue*

Summary

This command is used in conjunction with the FIELD command. A constant value of the variable given in the FIELD command is specified with this boundary condition.

Parameter	Value	Default
{=}	{= | are | is}	–
VariableValue	string	–

7.15 Equation Specification

Equation Commands

Ampere

Syntax

Eq Ampere for *Associated DoF* [*{of}* *SpeciesName* | *{in}* *MaterialPhaseName* | *{ls}* *{a | b | c}*] *{on}* *MeshPart* using *Interp* with *{Term_0}* . . . *{Term_N}*

Scope

Aria Region

Summary

Activates the *Ampere Equation*.

Description

The only admissible value for *Dof* is `VECTOR_POTENTIAL`.

Admissible values of *Interp* are $Q_1, Q_2, Q_{2S}, P_1, P_0$, for standard volume elements and $Q_1P_0, Q_2P_0, Q_{2S}P_0, Q_1Q_1$ and Q_1Q_2 for shell elements.

Admissible values of *Term_n* are `MASS`, `LUMPED_MASS`, `DIFF`, `ADV`, `SRC`, and `XFER`.

With the exception of `XFER` these terms are described in *Ampere Equation*.

The *Meshpart* must be an active element block.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Associated DoF	string	–
{on}	{@ | at | for | in | on | over}	–
MeshPart	string	–
Interp	string	–
{Term_n}	See **Description**	–

Brinkman_Momentum

Beta Capability
The porous media capability is under active development hence the Brinkman momentum equation should be used with caution.

Syntax

Eq Brinkman_Momentum for *Associated DoF* [*{of} SpeciesName* | *{in} MaterialPhaseName* | *{ls} {a | b | c}*] *{on} MeshPart* using *Interp* with *{Term_0}* ... *{Term_N}*

Scope

Aria Region

Summary

Activates the *Brinkman Momentum* equation.

Description

The only admissible value for *Dof* is `VELOCITY`.

Admissible values of *Interp* are Q_1, Q_2, Q_{2S}.

Admissible values of *Term_n* are `MASS`, `LUMPED_MASS`, `ADV`, `DIFF`, `SRC`, and `XFER`.

With the exception of `XFER` these terms are described in *Brinkman Momentum*.

The *Meshpart* must be an active element block.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Associated DoF	string	–
{on}	{@ | at | for | in | on | over}	–
MeshPart	string	–
Interp	string	–
{Term_n}	See **Description**	–

Charge_Density

Beta Capability
The charge density capability is not well tested and should be used with caution.

Syntax

Eq Charge_Density for *Associated DoF* [*{of}* *SpeciesName* | *{in}* *MaterialPhaseName* | *{ls}* *{a | b | c}*] *{on}* *MeshPart* using *Interp* with *{Term_0}* ... *{Term_N}*

Scope

Aria Region

Summary

Activates the *Charge Density Equation*.

Description

The only admissible value for *Dof* is `CHARGE_DENSITY`.

Admissible values of *Interp* are Q_1, Q_2, Q_{2S}.

Admissible values of *Term_n* are `MASS`, `LUMPED_MASS`, `ADV`, `DIFF`, and `XFER`.

With the exception of `XFER` these terms are described in *Charge Density Equation*.

The *Meshpart* must be an active element block.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Associated DoF	string	–
{on}	{@ | at | for | in | on | over}	–
MeshPart	string	–
Interp	string	–
{Term_n}	See **Description**	–

Continuity

Syntax

Eq Continuity for *Associated DoF* [*{of}* *SpeciesName* | *{in}* *MaterialPhaseName* | *{ls}* *{a | b | c}*] *{on}* *MeshPart* using *Interp* with *{Term_0}* . . . *{Term_N}*

Scope

Aria Region

Summary

Activates the *Conservation of Mass* equation.

Description

The only admissible value for *Dof* is `PRESSURE`.

Admissible values of *Interp* are $Q_1, Q_2, Q_{2S}, P_1, P_0$, for standard volume elements and $Q_1P_0, Q_2P_0, Q_{2S}P_0, Q_1Q_1$ and Q_1Q_2 for shell elements.

Admissible values of *Term_n* are `MASS`, `LUMPED_MASS`, `DIV`, `ADV`, `SRC`, and `XFER`.

With the exception of `XFER` these terms are described in *Conservation of Mass*.

The *Meshpart* must be an active element block.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Associated DoF	string	–
{on}	{@ | at | for | in | on | over}	–
MeshPart	string	–
Interp	string	–
{Term_n}	See **Description**	–

Current

Syntax

Eq Current for *Associated DoF* [*{of}* *SpeciesName* | *{in}* *MaterialPhaseName* | *{ls}* *{a | b | c}*] *{on}* *MeshPart* using *Interp* with *{Term_0}* . . . *{Term_N}*

Scope

Aria Region

Summary

Activates the *Current Equation*.

Description

The only admissible value for *Dof* is `VOLTAGE`.

Admissible values of *Interp* are Q_1, Q_2, Q_{2S}.

Admissible values of *Term_n* are `DIFF`, `SRC`, and `XFER`.

With the exception of `XFER` these terms are described in *Current Equation*.

The *Meshpart* must be an active element block.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Associated DoF	string	–
{on}	{@ | at | for | in | on | over}	–
MeshPart	string	–
Interp	string	–
{Term_n}	See **Description**	–

Energy

Syntax

Eq Energy for *Associated DoF* [*{of}* *SpeciesName* | *{in}* *MaterialPhaseName* | *{ls}* *{a | b | c}*] *{on}* *MeshPart* using *Interp* with *{Term_0}* ... *{Term_N}*

Scope

Aria Region

Summary

Activates the *Conservation of Energy* equation.

Description

Admissible values for *Dof* are `TEMPERATURE` and `ENTHALPY`.

Admissible values of *Interp* are Q_1, Q_2, Q_{2S} for standard volume elements and Q_1P_0, Q_2P_0, Q_1Q_1 and Q_1Q_2 for shell elements.

Admissible values of *Term_n* are `MASS`, `LUMPED_MASS`, `ADV`, `DIFF`, `SRC`, and `XFER`.

With the exception of `XFER` these terms are described in *Conservation of Energy*.

The *Meshpart* must be an active element block.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Associated DoF	string	–
{on}	{@ | at | for | in | on | over}	–
MeshPart	string	–
Interp	string	–
{Term_n}	See **Description**	–

Extension_Speed

Syntax

Eq Extension_Speed for *Associated DoF* [*{of}* *SpeciesName* | *{in}* *MaterialPhaseName* | *{ls}* *{a | b | c}*] *{on}* *MeshPart* using *Interp* with *{Term_0}* . . . *{Term_N}*

Scope

Aria Region

Summary

Activates an equation for extension speed, which is used by the level set equation (*Level Set/CDFEM*) to define the interface speed.

Description

The only admissible value for *Dof* is `EXTENSION_SPEED`.

Admissible values of *Interp* are Q_1, Q_2, Q_{2S}.

Admissible values of *Term_n* are `DEF`, `SRC`, and `XFER`.

With the exception of `XFER` these terms are described in *Generalized Conservation Equation*.

The *Meshpart* must be an active element block.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Associated DoF	string	–
{on}	{@ | at | for | in | on | over}	–
MeshPart	string	–
Interp	string	–
{Term_n}	See **Description**	–

Level_Set

Syntax

Eq Level_Set for *Associated DoF* [*{of}* *SpeciesName* | *{in}* *MaterialPhaseName* | *{ls}* *{a | b | c}*] *{on}* *MeshPart* using *Interp* with *{Term_0}* ... *{Term_N}*

Scope

Aria Region

Summary

Activates the Level Set equation. This equation is often accompanied by additional commands described in *Level Set/CDFEM*.

Description

The only admissible value for *Dof* is `LEVEL_SET`.

Admissible values of *Interp* are Q_1, Q_2, Q_{2S}.

Admissible values of *Term_n* are `MASS`, `LUMPED_MASS`, `ADV`, `DIFF`, `SRC`, and `XFER`.

With the exception of `XFER` these terms are described in *Generalized Conservation Equation*.

The *Meshpart* must be an active element block.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Associated DoF	string	–
{on}	{@ | at | for | in | on | over}	–
MeshPart	string	–
Interp	string	–
{Term_n}	See **Description**	–

Lubrication

Beta Capability
The lubrication modeling capability is under active development hence this equation should be used with caution.

Syntax

Eq Lubrication for *Associated DoF* [*{of}* *SpeciesName* | *{in}* *MaterialPhaseName* | *{ls}* *{a | b | c}*] *{on}* *MeshPart* using *Interp* with *{Term_0}* . . . *{Term_N}*

Scope

Aria Region

Summary

Activates the Reynolds' *Lubrication Equation*.

Description

The only admissible value for *Dof* is `LUBRICATION_PRESSURE`.

The only admissible value for *Interp* is Q_1P_0.

Admissible values of *Term_n* are `MASS`, `DIFF`, and `BOUND`.

With the exception of `XFER` these terms are described in *Lubrication Equation*.

For problems with fluid-structural interactions, the lubrication equation needs to be defined on all blocks, including 3-D blocks adjoining the shell lubrication region. In order to accommodate this, there is an additional term, `DEF`, which applies the equation $p = 0$ on adjoining 3-D blocks where this is used. The other terms should not be used on 3-D blocks. It is weighted so that it does not affect the residual on the shell blocks.

The *Meshpart* must be an active shell element block.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Associated DoF	string	–
{on}	{@ | at | for | in | on | over}	–
MeshPart	string	–
Interp	string	–
{Term_n}	See **Description**	–

Mass_Balance

Syntax

Eq Mass_Balance for *Associated DoF* [*{of}* *SpeciesName* | *{in}* *MaterialPhaseName* | *{ls}* *{a | b | c}*] *{on}* *MeshPart* using *Interp* with *{Term_0}* . . . *{Term_N}*

Scope

Aria Region

Summary

The mass balance equation for porous media. See general porous flow information in *Porous Flow Equations*.

Description

Admissible value for *Dof* are `PRESSURE`, `DENSITY`, `SATURATION`, and `MASS_FRACTION`.

Admissible values of *Interp* are Q_1, Q_2, Q_{2S}.

Admissible values of *Term_n* are `MASS`, `LUMPED_MASS`, `ADV`, `UPWIND_ADV`, `DIFF`, `SRC`, and `XFER`.

With the exception of `XFER` these terms are described in *Porous Flow Equations*.

The *Meshpart* must be an active element block.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Associated DoF	string	–
{on}	{@ | at | for | in | on | over}	–
MeshPart	string	–
Interp	string	–
{Term_n}	See **Description**	–

Mesh

Syntax

Eq Mesh for *Associated DoF* [*{of}* *SpeciesName* | *{in}* *MaterialPhaseName* | *{ls}* *{a | b | c}*] *{on}* *MeshPart* using *Interp* with *{Term_0}* . . . *{Term_N}*

Scope

Aria Region

Summary

Activates the pseudo-solid mesh equation (see *Conservation of Solid Momentum*).

Description

The only admissible value for *Dof* is `MESH_DISPLACEMENTS`.

Admissible values of *Interp* are Q_1, Q_2, Q_{2S}.

Admissible values of *Term_n* are `DIFF`, `REF_DIFF`, `TSTRAIN`, `SRC`, and `XFER`.

With the exception of `XFER` these terms are described in *Generalized Conservation Equation*.

If both the `MESH` and `SOLID` equations are defined in the same region, then the TALE mesh motion algorithm is activated automatically. In this case, the `MESH` equation must be defined on all blocks in the region. More details of the TALE method can be found in *Conservation of Solid Momentum*.

The *Meshpart* must be an active element block.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Associated DoF	string	–
{on}	{@ | at | for | in | on | over}	–
MeshPart	string	–
Interp	string	–
{Term_n}	See **Description**	–

Momentum

Beta Capability
The momentum equation is tested for low Reynolds numbers, $R_e < 10$ but should be used with caution beyond that range.

Syntax

Eq Momentum for *Associated DoF* [*{of}* *SpeciesName* | *{in}* *MaterialPhaseName* | *{ls}* *{a | b | c}*] *{on}* *MeshPart* using *Interp* with *{Term_0}* ... *{Term_N}*

Scope

Aria Region

Summary

Activates the *Conservation of Fluid Momentum* equation.

Description

The only admissible value for *Dof* is `VELOCITY`.

Admissible values of *Interp* are Q_1, Q_2, Q_{2S}.

Admissible values of *Term_n* are `MASS`, `LUMPED_MASS`, `ADV`, `DIFF`, `SRC`, and `XFER`.

With the exception of `XFER` these terms are described in *Conservation of Fluid Momentum*.

The *Meshpart* must be an active element block.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Associated DoF	string	–
{on}	{@ | at | for | in | on | over}	–
MeshPart	string	–
Interp	string	–
{Term_n}	See **Description**	–

Porous_Enthalpy

Syntax

Eq Porous_Enthalpy for *Associated DoF* [*{of}* *SpeciesName* | *{in}* *MaterialPhaseName* | *{ls}* *{a | b | c}*] *{on}* *MeshPart* using *Interp* with *{Term_0}* . . . *{Term_N}*

Scope

Aria Region

Summary

The enthalpy transport equation porous media. See general porous flow information in *Porous Flow Equations*.

Description

The only admissible value for *Dof* are `TEMPERATURE`.

Admissible values of *Interp* are Q_1, Q_2, Q_{2S}.

Admissible values of *Term_n* are `MASS`, `LUMPED_MASS`, `ADV`, `UPWIND_ADV`, `DIFF`, `SRC`, and `XFER`.

With the exception of `XFER` these terms are described in *Porous Flow Equations*.

The *Meshpart* must be an active element block.

Additionally, one can choose whether to integrate the advection term by parts by setting the flag `INTEGRATE ADVECTION BY PARTS = false` in the `POROUS_FLOW_OPTIONS` block of `SOLUTION_OPTIONS`. The default is true.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Associated DoF	string	–
{on}	{@ | at | for | in | on | over}	–
MeshPart	string	–
Interp	string	–
{Term_n}	See **Description**	–

Porous_Species

Syntax

Eq Porous_Species for *Associated DoF* [*{of} SpeciesName* | *{in} MaterialPhaseName* | *{ls} {a | b | c}*] *{on} MeshPart* using *Interp* with *{Term_0} ... {Term_N}*

Scope

Aria Region

Summary

The species transport equation porous media. See general porous flow information in *Porous Flow Equations*.

Description

Admissible value for *Dof* are `PRESSURE`, `SPECIES`, and `SPECIES_FRACTION`.

Admissible values of *Interp* are Q_1, Q_2, Q_{2S}.

Admissible values of *Term_n* are `MASS`, `LUMPED_MASS`, `ADV`, `UPWIND_ADV`, `DIFF`, `SRC`, and `XFER`.

With the exception of `XFER` these terms are described in *Porous Flow Equations*.

The *Meshpart* must be an active element block.

Additionally, one can choose whether to integrate the advection term by parts by setting the flag `INTEGRATE ADVECTION BY PARTS = false` in the `POROUS_FLOW_OPTIONS` block of `SOLUTION_OPTIONS`. The default is true.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Associated DoF	string	–
{on}	{@ | at | for | in | on | over}	–
MeshPart	string	–
Interp	string	–
{Term_n}	See **Description**	–

Potential

Syntax

Eq Potential for *Associated DoF* [*{of}* *SpeciesName* | *{in}* *MaterialPhaseName* | *{ls}* *{a | b | c}*] *{on}* *MeshPart* using *Interp* with *{Term_0}* ... *{Term_N}*

Scope

Aria Region

Summary

Activates the *Potential Projection Equation*.

Description

The only admissible value for *Dof* are `POTENTIAL`.

Admissible values of *Interp* are Q_1, Q_2, Q_{2S}.

Admissible values of *Term_n* are `DEF` and `SRC`, and both terms must be active for this equation to behave properly. The value used in `SRC` must be a separately defined source equation.

With the exception of `XFER` these terms are described in *Potential Projection Equation*.

The *Meshpart* must be an active element block.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Associated DoF	string	–
{on}	{@ | at | for | in | on | over}	–
MeshPart	string	–
Interp	string	–
{Term_n}	See **Description**	–

Projection

Syntax

Eq [*{Lumped_|Lagged_|Quasi_Implicit_}*]Projection for *Associated DoF* [*{of}* *SpeciesName* | *{in}* *MaterialPhaseName* | *{ls}* *{a | b | c}*] *{on}* *MeshPart* using *Interp* with *{Term_0}* . . . *{Term_N}*

Scope

Aria Region

Summary

Perform a projection of an expression from integration points to nodal values.

Description

The string *Dof* refers to a new degree of freedom to be created and solved for via the projection equation. *Dof* can be chosen to be any expression valid at the integration points, in which case the string used for *Dof* should be the decorated version of the original expression name. A prefix of `PROJECTED_`, `PROJECTED_VECTOR_`, and `PROJECTED_TENSOR_` should be used for scalar, vector, and tensor expressions, respectively.

By default, projection equations use a consistent source term resulting in a formal L2 projection; however, projection equations can also use a lumped source term by specifying `LUMPED_` before `PROJECTION`. Projection equations are normally solved by the linear solver; however, they can be lagged and solved at the start of a time step by specifying `LAGGED_` or once per nonlinear iteration by specifying `QUASI_IMPLICIT_`. Using either the `LAGGED_` or `QUASI_IMPLICIT_` prefixes will not burden the linear solver, but explicit solution necessarily requires that both options use a lumped projection.

Admissible values of *Interp* are Q_1, Q_2, Q_{2S}.

The only admissible value of *Term_n* is `DEF`.

The *Meshpart* must be an active element block.

Example

To project the heat conduction on every block using linear elements, one would use the command

```
Eq Projection For Vector_Heat_Conduction On All_Blocks Using␣
↪Q1 With Def
```

Likewise, to project a mesh stress, one could specify

```
Eq Projection For Projected_Tensor_Mesh_Stress On All_Blocks␣
↪Using Q1 With Def
```

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Associated DoF	string	–
{on}	{@ | at | for | in | on | over}	–
MeshPart	string	–
Interp	string	–
{Term_n}	See **Description**	–

Radiation_Transport

Syntax

Eq Radiation_Transport for *Associated DoF* [*{of}* *SpeciesName* | *{in}* *MaterialPhaseName* | *{ls}* *{a | b | c}*] *{on}* *MeshPart* using *Interp* with *{Term_0}* . . . *{Term_N}*

Scope

Aria Region

Summary

Activates the *Radiation Transport Equation*.

Description

This enables the radiation transport equation in the specified mesh part. Note that the solution of this equation occurs in a separate PMR application coupled to Aria through MPMD and this equation line simply indicates that the specified mesh part be sent to the PMR application. An MPMD coupling block in the Aria region is required as well to turn on the *MPMD Coupling*.

The only admissible value for *Dof* are `RTE_SCALAR_FLUX`.

Admissible values of *Interp* are Q_1, Q_2, Q_{2S}. This input is ignored as this equation is not solved in Aria.

The only admissible value of *Term_n* is `XFER`.

The *Meshpart* must be an active element block.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Associated DoF	string	–
{on}	{@ | at | for | in | on | over}	–
MeshPart	string	–
Interp	string	–
{Term_n}	See **Description**	–

Solid

Beta Capability
Material models for this capability are under active development hence this equation should be used with caution.

Syntax

Eq Solid for *Associated DoF* [*{of} SpeciesName* | *{in} MaterialPhaseName* | *{ls} {a | b | c}*] *{on} MeshPart* using *Interp* with *{Term_0}* . . . *{Term_N}*

Scope

Aria Region

Summary

Activates the *Conservation of Solid Momentum* equation.

Description

The only admissible value for *Dof* is `SOLID_DISPLACEMENTS`.

Admissible values of *Interp* are Q_1, Q_2, Q_{2S}.

Admissible values of *Term_n* are `DIFF`, `TSTRAIN`, `SRC`, and `XFER`.

With the exception of `XFER` these terms are described in *Generalized Conservation Equation*.

If both the MESH and SOLID equations are defined in the same region, then the TALE mesh motion algorithm is activated automatically. In this case, the MESH equation must be defined on all blocks in the region. More details of the TALE method can be found in *Conservation of Solid Momentum*.

The *Meshpart* must be an active element block.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Associated DoF	string	–
{on}	{@ | at | for | in | on | over}	–
MeshPart	string	–
Interp	string	–
{Term_n}	See **Description**	–

Sp

Beta Capability
The spherical harmonics capability is not well tested and should be used with caution.

Syntax

Eq Sp *Order* for *Associated DoF* [*{of} SpeciesName* | *{in} MaterialPhaseName* | *{ls} {a | b | c}*] *{on} MeshPart* using *Interp* with *{Term_0}* . . . *{Term_N}*

Scope

Aria Region

Summary

Specifies the Spherical Harmonics equation set to solve for Participating Media Radiation (PMR). User provides the order, degree of freedom associated with the equation, the element type and the mesh part (element block) on which to apply the equation (see *SPn Radiation*).

Description

Admissible values for *Order* are odd values (1, 3, 5, . . .).

The only admissible value for *Dof* is ANGULAR_INTENSITY.

Admissible values of *Interp* are Q_1, Q_2, Q_{2S}.

The only admissible value for *Term_n* is SRC.

The *Meshpart* must be an active element block.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Associated DoF	string	–
{on}	{@ | at | for | in | on | over}	–
MeshPart	string	–
Interp	string	–
{Term_n}	See **Description**	–

Species

Beta Capability
The species equation is tested for low Reynolds numbers, $R_e < 10$ but should be used with caution beyond that range.

Syntax

Eq Species for *Associated DoF* [*{of} SpeciesName* | *{in} MaterialPhaseName* | *{ls} {a | b | c}*] *{on} MeshPart* using *Interp* with *{Term_0}* ... *{Term_N}*

Scope

Aria Region

Summary

Activates the *Conservation of Chemical Species* equation.

Description

The only admissible value for *Dof* is SPECIES.

Admissible values of *Interp* are Q_1, Q_2, Q_{2S}.

Admissible values of *Term_n* are MASS, LUMPED_MASS, ADV, DIFF, SRC, FRACBAL and XFER. With the exception of XFER these terms are described in *Conservation of Chemical Species*. The FRACBAL term may not be included with other terms.

Some other things to note about species equations in Aria:

- The FRACBAL term may be assigned to any species number.

- Species numbers in Aria are arbitrary; they may start at any value and need not be continuous.
- For the species fraction balances, Aria will automatically detect all species that are present in the problem and include them in the balance.

The *Meshpart* must be an active element block.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Associated DoF	string	–
{on}	{@ | at | for | in | on | over}	–
MeshPart	string	–
Interp	string	–
{Term_n}	See **Description**	–

Stress_Tensor_Projection

Syntax

Eq Stress_Tensor_Projection for *Associated DoF* [*{of}* *SpeciesName* | *{in}* *MaterialPhaseName* | *{ls}* *{a | b | c}*] *{on}* *MeshPart* using *Interp* with *{Term_0}* ... *{Term_N}*

Scope

Aria Region

Summary

Activates the Stress Tensor Projection equation.

Description

The only admissible value for *Dof* is `STRESS_TENSOR`.

Admissible values of *Interp* are Q_1, Q_2, Q_{2S}.

The only admissible value for *Term_n* are `DEF`.

With the exception of `XFER` these terms are described in *Generalized Conservation Equation*.

The *Meshpart* must be an active shell element block.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Associated DoF	string	–
{on}	{@ | at | for | in | on | over}	–
MeshPart	string	–
Interp	string	–
{Term_n}	See **Description**	–

Suspension

Beta Capability
The suspension modeling capability is not well tested and should be used with caution.

Syntax

Eq Suspension for *Associated DoF* [*{of}* *SpeciesName* | *{in}* *MaterialPhaseName* | *{ls}* *{a | b | c}*] *{on}* *MeshPart* using *Interp* with *{Term_0}* ... *{Term_N}*

Scope

Aria Region

Summary

Activates the *Suspension Equation*.

Description

The only admissible value for *Dof* is `PHI`.

Admissible values of *Interp* are Q_1, Q_2, Q_{2S}.

Admissible values of *Term_n* are `ADV`, `DIFF`, `SRC`, and `XFER`.

With the exception of `XFER` these terms are described in *Suspension Equation*.

The *Meshpart* must be an active element block.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Associated DoF	string	–
{on}	{@ | at | for | in | on | over}	–
MeshPart	string	–
Interp	string	–
{Term_n}	See **Description**	–

Voltage

Syntax

Eq Voltage for *Associated DoF* [*{of} SpeciesName* | *{in} MaterialPhaseName* | *{ls} {a | b | c}*] *{on} MeshPart* using *Interp* with *{Term_0}* ... *{Term_N}*

Scope

Aria Region

Summary

Activates the *Voltage Equation* (electric-displacement formulation). See also the *Current Equation*.

Description

The only admissible value for *Dof* is `VOLTAGE`.

Admissible values of *Interp* are Q_1, Q_2, Q_{2S}.

Admissible values of *Term_n* are `DIFF`, `SRC`, and `XFER`.

With the exception of `XFER` these terms are described in *Voltage Equation*.

The *Meshpart* must be an active element block.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Associated DoF	string	–
{on}	{@ | at | for | in | on | over}	–
MeshPart	string	–
Interp	string	–
{Term_n}	See **Description**	–

Additional Line Commands

Elasticity Formulation

Syntax

Elasticity Formulation = *{PLANE STRESS | PLANE STRAIN}*

Scope

Aria Region

Summary

Assigns the elasticity formulation type for two-dimensional problems involving the `MESH` and `SOLID` equations (see *Conservation of Solid Momentum*).

Pressure Stabilization

Syntax

Pressure Stabilization IS *Type* WITH SCALING = *C*

Scope

Aria Region

Summary

Prescribe a stabilization technique for solving *Momentum* and *Continuity* with equal order interpolation.

Description

Aria supports both PSPG (Pressure Stabilized Petrov-Galerkin) and PSPP (Pressure Stabilized Pressure Projection) stabilization techniques for solving the `MOMENTUM` and `CONTINUITY` equations with equal order interpolation.

Valid options for the *Type* specification are:

- NO_STABILIZATION
- PSPG_CONSTANT
- PSPG_LOCAL
- PSPG_GLOBAL
- PSPP_CONSTANT

NO_STABILIZATION disables any stabilization.

PSPP_CONSTANT results in the recently developed stabilization technique of [59] and [60].

In the PSPG forms of stabilization, introduced by [61], terms from the momentum equation are added to the continuity equation scaled by a multiplier, α. The exact form of the multiplier depend on a global Reynolds number that is defined as

$$Re \equiv \frac{\rho|\boldsymbol{v}|\langle h\rangle}{2\mu} \tag{7.1}$$

Here, ρ is the density, μ is the viscosity, $|\boldsymbol{v}|$ is a velocity scale and $\langle h\rangle$ is an element length scale. Armed with Re, the stabilization multiplier α is defined in one of two ways.

$$\alpha \equiv \begin{cases} \frac{\tau\langle h\rangle^2}{12\mu} & Re \leq 3 \\ \frac{\tau\langle h\rangle}{2\rho|\boldsymbol{v}|} & Re > 3 \end{cases} \tag{7.2}$$

NB: Currently, Aria always uses the low-Reynolds number form of α.

The PSPG_LOCAL method computes $|\boldsymbol{v}|$ and $\langle h\rangle$ within each element. The PSPG_GLOBAL method computes $|\boldsymbol{v}|$ and $\langle h\rangle$ as averages over all of the elements with the MOMENTUM equation defined. The PSPG_CONSTANT gives $|\boldsymbol{v}|$ and $\langle h\rangle$ a value of 1 (one) and just uses the scale factor.

Parameter	Value	Default
Type	See **Description**	–
C	real	–

Save Residuals

Syntax

Save Residuals = *{OFF | BEFORE_BCS | AFTER_BCS}*

Scope

Aria Region

Summary

Causes Aria to save the residuals to a field with the prefix `residual->`, e.g., `residual->Temperature`. This will be done for all fields (though we could make it a per-field option).

For the choice `BEFORE_BCS` the residuals will be saved at the point in the assembly process where the primary equations have been assembled but prior to the assembly of any boundary conditions or distinguishing conditions. For the choice of `AFTER_BCS` the residuals will be saved after all BCs and distinguishing conditions have been applied. The default, `OFF`, is to not save the residuals.

This feature is only applicable when using the `NEWTON` nonlinear solution strategy.

Integration Rule

Beta Capability
Non-default integration rules are not heavily tested and should be used with caution.

Syntax

Integration Rule for *Block Name* = *Order*

Scope

Aria Region

Summary

Overrides the default integration rule for the equations defined on *Block Name*.

Parameter	Value	Default
Block Name	string	–
Order	integer	–

Advection Velocity

Syntax

Advection Velocity for *Equation Name* = *Vel*

Scope

Aria Region

Summary

Defines the velocity model to be used for the equation *Equation Name*. Velocity model can be `XFER` or any other vector velocity model defined in the Region.

Parameter	Value	Default
Block Name	string	–
Vel	string	–

7.16 Initial Conditions

Generic Ic Models

A list of available initial conditions is provided below

Calore_User_Sub

Syntax

Ic For *DofName* ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshExtent* = Calore_User_Sub ([Power_Output = *power_output*] | [Flux_Output = *flux_output*] | [Toggle = *toggle*] | Name = *name* | Type = *type* | [Multiplier = *multiplier*] | [Material_Data_Block = *material_data_block*] | [Data = *data*])

Summary

Values from a Calore user subroutine

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output

If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Flux_Output

If specified, the average flux of the given BC/Src is calculated and stored into the global variable name specified as the *flux_output* argument.

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
DofName	string	–
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshExtent	string	–
power_output	"string"	–
flux_output	"string"	–
toggle	"string"	–
name	"string"	–
type	"string"	–
multiplier	real	1
material_data_block	"string"	–
data	"string"	–

Cantera

Syntax

Ic For *DofName* ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshExtent* = Cantera ([Power_Output = *power_output*] | [Flux_Output = *flux_output*] | [Toggle = *toggle*] | [Omit_Fd_Sens = *omit_fd_sens*])

Summary

Cantera enthalpy computed from temperature

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output

If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Flux_Output

If specified, the average flux of the given BC/Src is calculated and stored into the global variable name specified as the *flux_output* argument.

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Omit_Fd_Sens

Omits calculating this sensitivity when using FD sensitivities when `true`.

Parameter	Value	Default
DofName	string	–
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshExtent	string	–
power_output	"string"	–
flux_output	"string"	–
toggle	"string"	–
omit_fd_sens	"string"	false

Cht_Robin

Syntax

Ic For *DofName* ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshExtent* = Cht_Robin ([Power_Output = *power_output*] | [Flux_Output = *flux_output*] | [Toggle = *toggle*] | Temperature_Field = *temperature_field*)

Summary

Dirichlet boundary condition for interface coupled with Robin style BC CHT_ROBIN. This BC computes the heat flux for the other side.

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output

If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Flux_Output

If specified, the average flux of the given BC/Src is calculated and stored into the global variable name specified as the *flux_output* argument.

Toggle

If given, specifies which *toggle block* controls whether this BC is

active or not.

Parameter	Value	Default
DofName	string	–
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshExtent	string	–
power_output	"string"	–
flux_output	"string"	–
toggle	"string"	–
temperature_field	"string"	–

Constant

Syntax

Ic For *DofName* ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshExtent* = Constant ([Power_Output = *power_output*] | [Flux_Output = *flux_output*] | [Toggle = *toggle*] | Value = *value*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output

If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Flux_Output

If specified, the average flux of the given BC/Src is calculated and stored into the global variable name specified as the *flux_output* argument.

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
DofName	string	–
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshExtent	string	–
power_output	"string"	–
flux_output	"string"	–
toggle	"string"	–
value	real	–

Copied

Syntax

Ic For *DofName* ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshExtent* = Copied ([Power_Output = *power_output*] | [Flux_Output = *flux_output*] | [Toggle = *toggle*] | Source = *source*)

Summary

Copied value from another expression

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output

If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Flux_Output

If specified, the average flux of the given BC/Src is calculated and stored into the global variable name specified as the *flux_output* argument.

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
DofName	string	–
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshExtent	string	–
power_output	"string"	–
flux_output	"string"	–
toggle	"string"	–
source	"string"	–

Darcy

Syntax

Ic For *DofName* ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshExtent* = Darcy ([Gx = *gx*] | [Gy = *gy*] | [Gz = *gz*] | [Relative_To = *relative_to*])

Summary

Provides the Darcy velocity model.

Description

For phase p, the velocity provided is

$$\mathbf{u}_p = \phi S_p \mathbf{u}_{matrix} - \frac{k_{r,p}}{\mu_p} \mathbf{K}_s \left(\nabla P_p - \rho_p \mathbf{g} \right),$$

where $\mathbf{u}_{matrix}$ is the optional matrix velocity as specified in the `relative to` parameter, ϕ is the porosity, S_p is the optional phase saturation, $k_{r,p}$ is the relative permeability of the phase, μ_p is the dynamic viscosity of the phase, $\mathbf{K}_s$ is the intrinsic permeability tensor, P_p is the phase pressure, ρ_p is the density of the phase, and $\mathbf{g}$ is the gravitational acceleration vector.

If the optional phase saturation S_p is provided, the pore volume is assumed to be occupied by multiple phases (e.g., liquid / gas) in accordance to their saturations, otherwise it is occupied fully by a single phase. Note, the sum of phase saturation should equal to one i.e., individual phase volumes should sum to the total pore volume.

Additionally, if the matrix is deforming, the optional `relative to` parameter should specify which phase should be used as the matrix velocity. For instance, `relative to = solid_phase` would ensure the velocity due to the pressure gradient is relative to the deforming solid matrix.

Lastly, note the Darcy velocity is the superficial velocity and is related to the pore scale velocity through the product of porosity and optional phase saturation.

Parameter	Value	Default
DofName	string	–
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshExtent	string	–
gx	real	0
gy	real	0
gz	real	0
relative_to	"string"	–

Darcy_Solvent

Syntax

Ic For *DofName* ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshExtent* = Darcy_Solvent ([Gx = *gx*] | [Gy = *gy*] | [Gz = *gz*] | [Relative_To = *relative_to*])

Summary

This model is intended for problems where the porous species equation is being solved for pressure with a solvent species concentration as the mass term in order to solve for the velocity of that solvent species.

Parameter	Value	Default
DofName	string	–
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshExtent	string	–
gx	real	0
gy	real	0
gz	real	0
relative_to	"string"	–

Dirichlet_From_Temperature

Syntax

Ic For *DofName* ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshExtent* = Dirichlet_From_Temperature ([Temperature = *temperature*] | [Pressure = *pressure*] | [Mass_Fraction_0 = *mass_fraction_0*] | [Mass_Fraction_1 = *mass_fraction_1*] | [Mass_Fraction_2 = *mass_fraction_2*] | [Mass_Fraction_3 = *mass_fraction_3*] | [Mass_Fraction_4 = *mass_fraction_4*] | [Mass_Fraction_5 = *mass_fraction_5*] | [Mass_Fraction_6 = *mass_fraction_6*] | [Mass_Fraction_7 = *mass_fraction_7*] | [Mass_Fraction_8 = *mass_fraction_8*])

Summary

Enthalpy computed from mass fraction, temperature, and pressure

Parameter	Value	Default
DofName	string	–
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshExtent	string	–
temperature	real	0
pressure	real	0
mass_fraction_0	real	0
mass_fraction_1	real	0
mass_fraction_2	real	0
mass_fraction_3	real	0
mass_fraction_4	real	0
mass_fraction_5	real	0
mass_fraction_6	real	0
mass_fraction_7	real	0
mass_fraction_8	real	0

Encore_Saturation

Syntax

Ic For *DofName* ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshExtent* = Encore_Saturation (Name = *name* | [Result_Name = *result_name*] | [Eval_Type = *eval_type*])

Summary

A Van Genuchten model for air/water capillary pressure based on an encore function for saturation

Parameter	Value	Default
DofName	string	–
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshExtent	string	–
name	"string"	–
result_name	"string"	–
eval_type	"string"	–

Exponential

Syntax

Ic For *DofName* ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshExtent* = Exponential ([Power_Output = *power_output*] | [Flux_Output = *flux_output*] | [Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output

If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Flux_Output

If specified, the average flux of the given BC/Src is calculated and stored into the global variable name specified as the *flux_output* argument.

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
DofName	string	–
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshExtent	string	–
power_output	"string"	–
flux_output	"string"	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Fortran_User_Sub

Syntax

Ic For *DofName* ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshExtent* = Fortran_User_Sub ([Power_Output = *power_output*] | [Flux_Output = *flux_output*] | [Toggle = *toggle*] | [Multiplier = *multiplier*] | Name = *name* | Type = *type* | [Real_Data = *real_data*] | [Int_Data = *int_data*] | [Resource_Name = *resource_name*] | [Data = *data*] | [Material_Data_Block = *material_data_block*])

Summary

Dirichlet value from a Fortran user subroutine

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output

If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Flux_Output

If specified, the average flux of the given BC/Src is calculated and

stored into the global variable name specified as the *flux_output* argument.

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
DofName	string	–
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshExtent	string	–
power_output	“string”	–
flux_output	“string”	–
toggle	“string”	–
multiplier	real	1
name	“string”	–
type	“string”	–
real_data	“string”	–
int_data	“string”	–
resource_name	“string”	–
data	“string”	–
material_data_block	“string”	–

From_Chemeq

Syntax

Ic For *DofName* ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshExtent* = From_Chemeq ()

Summary

Species fraction of one species extracted from ChemEq

Parameter	Value	Default
DofName	string	–
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshExtent	string	–

From_Mass_Fraction

Syntax

Ic For *DofName* ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshExtent* = From_Mass_Fraction ()

Summary

Species fraction from mass fraction

Parameter	Value	Default
DofName	string	–
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshExtent	string	–

From_Mesh_Displacement

Syntax

Ic For *DofName* ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshExtent* = From_Mesh_Displacement ()

Parameter	Value	Default
DofName	string	–
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshExtent	string	–

From_No_Material_Phase

Syntax

Ic For *DofName* ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshExtent* = From_No_Material_Phase ([Power_Output = *power_output*] | [Flux_Output = *flux_output*] | [Toggle = *toggle*])

Summary

Copy the temperature from NO_MATERIAL_PHASE to the desired material phase.

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output

If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Flux_Output

If specified, the average flux of the given BC/Src is calculated and stored into the global variable name specified as the *flux_output* argument.

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
DofName	string	–
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshExtent	string	–
power_output	“string”	–
flux_output	“string”	–
toggle	“string”	–

From_Other_Phase

Syntax

Ic For *DofName* ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshExtent* = From_Other_Phase ()

Summary

Get the saturation from one minus the other phase saturation

Parameter	Value	Default
DofName	string	–
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshExtent	string	–

From_Species

Syntax

Ic For *DofName* ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshExtent* = From_Species ()

Summary

Species fraction of one species from it's concentration and the overall concentration

Parameter	Value	Default
DofName	string	–
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshExtent	string	–

From_Temperature

Syntax

Ic For *DofName* ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshExtent* = From_Temperature (*{t | value}* = *t* | [Poffset = *poffset*])

Summary

Cantera enthalpy computed from temperature

Parameter	Value	Default
DofName	string	–
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshExtent	string	–
t	real	–
poffset	real	0

Global

Syntax

Ic For *DofName* ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshExtent* = Global ([Power_Output = *power_output*] | [Flux_Output = *flux_output*] | [Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output

If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Flux_Output

If specified, the average flux of the given BC/Src is calculated and stored into the global variable name specified as the *flux_output* argument.

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
DofName	string	–
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshExtent	string	–
power_output	"string"	–
flux_output	"string"	–
toggle	"string"	–
global_name	"string"	–
component	integer	0

Mass_Average

Syntax

Ic For *DofName* ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshExtent* = Mass_Average ()

Summary

Enthalpy computed from mass average of species enthalpies

Description

Defines a mixture enthalpy from mass fractions as

$$h = \frac{\sum_i Y_i h_i}{\sum_i Y_i}$$

Parameter	Value	Default
DofName	string	–
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshExtent	string	–

Melting_Capillary_Darcy

Syntax

Ic For *DofName* ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshExtent* = Melting_Capillary_Darcy ([Melt_Dt = *melt_DT*] | [Melt_Time = *melt_time*] | [Gx = *gx*] | [Gy = *gy*] | [Gz = *gz*])

Parameter	Value	Default
DofName	string	–
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshExtent	string	–
melt_DT	real	5
melt_time	real	1.798e+308
gx	real	0
gy	real	0
gz	real	0

Nd_Table

Syntax

Ic For *DofName* ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshExtent* = Nd_Table ([Power_Output = *power_output*] | [Flux_Output = *flux_output*] | [Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output

If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Flux_Output

If specified, the average flux of the given BC/Src is calculated and stored into the global variable name specified as the *flux_output* argument.

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
DofName	string	–
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshExtent	string	–
power_output	“string”	–
flux_output	“string”	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Ic For *DofName* ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshExtent* = Polynomial ([Power_Output = *power_output*] | [Flux_Output = *flux_output*] | [Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output
If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Flux_Output
If specified, the average flux of the given BC/Src is calculated and stored into the global variable name specified as the *flux_output* argument.

Toggle
If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
DofName	string	–
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshExtent	string	–
power_output	"string"	–
flux_output	"string"	–
toggle	"string"	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Ic For *DofName* ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshExtent* = Scalar_String_Function ([Power_Output = *power_output*] | [Flux_Output = *flux_output*] | [Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output

If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Flux_Output

If specified, the average flux of the given BC/Src is calculated and stored into the global variable name specified as the *flux_output* argument.

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
DofName	string	–
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshExtent	string	–
power_output	“string”	–
flux_output	“string”	–
toggle	“string”	–
f	“string”	–

Schloegl

Syntax

Ic For *DofName* ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshExtent* = Schloegl (K_Phi = *k_phi* | F = *f* | [Gx = *gx*] | [Gy = *gy*] | [Gz = *gz*] | [Proton_Name = *proton_name*])

Parameter	Value	Default
DofName	string	–
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshExtent	string	–
k_phi	real	–
f	real	–
gx	real	0
gy	real	0
gz	real	0
proton_name	“string”	Hp

User_Field

Syntax

Ic For *DofName* ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshExtent* = User_Field ([Power_Output = *power_output*] | [Flux_Output = *flux_output*] | [Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output

If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Flux_Output

If specified, the average flux of the given BC/Src is calculated and

stored into the global variable name specified as the *flux_output* argument.

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
DofName	string	–
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshExtent	string	–
power_output	"string"	–
flux_output	"string"	–
toggle	"string"	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Ic For *DofName* ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshExtent* = User_Function ([Power_Output = *power_output*] | [Flux_Output = *flux_output*] | [Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output

If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Flux_Output

If specified, the average flux of the given BC/Src is calculated and

stored into the global variable name specified as the *flux_output* argument.

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
DofName	string	–
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshExtent	string	–
power_output	"string"	–
flux_output	"string"	–
toggle	"string"	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Ic For *DofName* ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshExtent* = User_Plugin ([Power_Output = *power_output*] | [Flux_Output = *flux_output*] | [Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output

If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Flux_Output

If specified, the average flux of the given BC/Src is calculated and

stored into the global variable name specified as the *flux_output* argument.

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
DofName	string	–
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshExtent	string	–
power_output	"string"	–
flux_output	"string"	–
toggle	"string"	–
Name	"string"	–
plugin_parameters	"string"	–

Van_Genuchten_Bc

Syntax

Ic For *DofName* ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshExtent* = Van_Genuchten_Bc Wetting_Saturation_Expression_Name = *wetting_saturation_expression_name*

Summary

A Van Genuchten model for air/water capillary pressure based on an arbitrary function for wetting phase saturation

Parameter	Value	Default
DofName	string	–
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshExtent	string	–
wetting_saturation_expression_name	"string"	–

Vector_String_Function

Syntax

Ic For *DofName* ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshExtent* = Vector_String_Function (F_X = *f_x* | F_Y = *f_y* | [F_Z = *f_z*])

Summary

Value from a string function

Parameter	Value	Default
DofName	string	–
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshExtent	string	–
f_x	“string”	–
f_y	“string”	–
f_z	“string”	–

Wall_Function

Syntax

Ic For *DofName* ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshExtent* = Wall_Function Wall_Friction_Factor = *wall_friction_factor*

Summary

Turbulent kinetic energy using wall function

Parameter	Value	Default
DofName	string	–
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshExtent	string	–
wall_friction_factor	real	–

Ic Input_Output_Region

Syntax

Ic Input_Output_Region *{on}* *MeshPartIdentifier* *FieldVariableName* ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*]) *{=}* *FileVariableName* [Region *{=}* *RegionName*]

Summary

Uses a variable stored on an INPUT_OUTPUT region to initialize a field. Unlike the FILE_READ IC, this IC allows the user to use an variable on a different mesh to initialize a field. It sets up the interpolation transfers needed to interpolate the field from the source mesh (specified in the INPUT_OUTPUT_REGION) to the destination mesh.

Parameter	Value	Default
{on}	{@ | at | for | in | on | over}	–
MeshPartIdentifier	string	–
FieldVariableName	string	–
{of}	{of | species}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{=}	{= | are | is}	–
FileVariableName	string	–

Ic Read_File

Syntax

Ic Read_File *dof* ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*]) *{=}* *FileVariable* [At Time *Value*]

Summary

This IC command will initialize the chosen Aria DOF with values from the field with the name given by a variable selected from the input mesh database. Only initialization of nodal variables with nodal variables from another file is currently supported.

Description

This IC command will initialize the field DOF in the current analysis with values from the input mesh database FileVariable. For example, if the mesh database as specified in the FINITE ELEMENT MODEL command block contains a vector field named U then one could use the file to initialize a velocity field by setting DOF to VELOCITY and the FileVariable argument to U.

In cases where the field DOF has a several components (i.e. DOF is a vector field) and the input mesh database contains the vector components defined as scalar fields, the command must be called once for each component. Here the DOF is specified as DOF_component.

A typical usage example would be initialize a transient calculation from a previously performed steady-state calculation in the following way. The steady-state calculation would be performed and its output variables saved to some output file. In most cases one might employ a copy of the output file to be used as a mesh file for the transient calculation. Finally, one would use the file IC command line to tie the variable named "FileVariable" in the mesh file to the variable named DOF in the current simulation.

When the mesh database file contains multiple timesteps and the TIME option is not supplied, the default behavior will be to extract the requested field from the last time plane in the file. The TIME option must be used if the initial condition is to be extracted from any specific database time plane other than the last time plane. In using the TIME option the database time step nearest to the specified time value will be used for the assignment. For an initial condition taken from a steady-state results file, the optional TIME parameter should be omitted.

Steady solution results are sometimes written to file in two steps. When this command is used to initialize values from a steady solution one must select the solution file step according to the content of the steady solution file results. If the file contains two solution steps then one should employ the time option and select the time corresponding to the second step.

CAVEAT: Current framework services do not allow restricting this initialization by mesh subsets. e. g. If one wishes to initialize the temperature, then all nodes in the mesh will be initialized.

NOTE: Those wishing to perform initializations as part of a formal restart are referred to RESTART DATA services.

Parameter	Value	Default
dof	string	–
{of}	{of | species}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{=}	{= | are | is}	–
FileVariable	string	–

Legacy IC Models

Initial Condition

Scope

Aria Region

Summary

Allows the specification of a user-defined initial conditions on any combination of volumes, surfaces and nodes.

Description

There are three methods for setting the initial conditions of the thermal model. The temperature specification must be of exactly one type: e.g., constant, defined in a user subroutine, or read from a solution database file. If a user subroutine is used, real and integer data may be declared that will be local in scope to this instance of the boundary condition. This data may be accessed using a user query function.

```
begin Initial Condition BlockName

   Add Surface PartList...

   Add Volume PartList...

   All Volumes

   Node Subroutine {=} Name

   Temperature {=} t0

   Use Data Block Name

   Use File Variable File_variable_name For Aria_variable_name ([{of}␣
↪SpeciesName]  | [{in} MaterialPhaseName]) [ At Time Value  ]

end Initial Condition BlockName
```

Line Commands

Add Surface

Syntax

Add Surface *PartList. . .*

Summary

Surface name to which this initial condition applies. May specify a space-delimited list.

Parameter	Value	Default
PartList	string. . .	–

Add Volume

Syntax

Add Volume *PartList. . .*

Summary

Block names to which this initial condition applies. May specify a space-delimited list.

Parameter	Value	Default
PartList	string. . .	–

All Volumes

Syntax

All Volumes

Summary

Apply this command block to all volumes in the mesh file.

Node Subroutine

Syntax

Node Subroutine *{=}* *Name*

Summary

Specifies that the named user-defined subroutine be used. At most one subroutine name may be specified for a given quantity. This subroutine must conform to the “node signature” argument list type.

Parameter	Value	Default
{=}	{= | are | is}	–
Name	string	–

Temperature

Syntax

Temperature *{=}* *t0*

Summary

Set the initial temperature to a constant value.

Parameter	Value	Default
{=}	{= | are | is}	–
t0	real	–

Use Data Block

Syntax

Use Data Block *Name*

Summary

Reference to predefined data to be used by the user subroutine. These values may be changed by the user subroutine.

Parameter	Value	Default
Name	string	–

Use File Variable

Syntax

Use File Variable *File_variable_name* For *Aria_variable_name* ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*]) [At Time *Value*]

Summary

Assign file nodal variable to valid registered nodal variable for initial condition.

Description

This line command specifies that the named Aria variable be initialized from a variable contained in the mesh file that is assigned to the region. For example, this line command may be used to initialize a transient calculation from a previously performed steady-state calculation in the following way. Perform the steady-state calculation and output the temperature. Then copy that output file and use it as a mesh file for the transient calculation. Finally, use this line command to tie the variable named `file_variable_name` in the mesh file to the aria variable named `aria_variable_name`. Here `aria_variable_name` must be set to `nonlinear_solution->temperature`.

The closest time step to the time value found the IO database will be used for the assignment. For an initial condition taken from a steady state results file, omit the optional time command. The optional command is necessary if the initial condition is coming from a file with multiple timesteps.

Note:

1. Current framework services do not allow restricting this initialization by mesh subset. e.g., if you want to initialize the temperature, then all nodes in the mesh will be initialized.
2. This command line syntax only supports initialization of temperature field. This restriction may be removed in the future so that chemistry variables or user-defined variables may also be initialized.
3. Sierra Mechanics uses a DOF naming convention which distinguishes vector fields from scalar fields using an underscore plus component notation (e.g. `field_variable_base_name_1`, `field_variable_base_name_X`). Hence use of external scalar field variable names that include an indexed notation will require that the file variable be renamed in order to be processed correctly as a scalar field in the `IC READ_FILE` setup.

Parameter	Value	Default
File_variable_name	string	–
Aria_variable_name	string	–
{of}	{of | species}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–

Ic Circ_X

Syntax

Ic Circ_X *{on} MeshExtentIdentifier FieldVariableName* ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*]) *Value_type {=} Angular_velocity. . .*

Summary

Spatially dependent circular x Dirichlet initial condition valid in an Aria region. BC depends upon its coordinate.

Parameter	Value	Default
{on}	{@ | at | for | in | on | over}	–
MeshExtentIdentifier	string	–
FieldVariableName	string	–
{of}	{of | species}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
Value_type	string	–
{=}	{= | are | is}	–
Angular_velocity	real. . .	–

Ic Circ_Y

Syntax

Ic Circ_Y *{on}* *MeshExtentIdentifier* *FieldVariableName* ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*]) *Value_type* *{=}* *Angular_velocity. . .*

Summary

Spatially dependent circular y Dirichlet initial condition valid in an Aria region. BC depends upon its coordinate.

Parameter	Value	Default
{on}	{@ | at | for | in | on | over}	–
MeshExtentIdentifier	string	–
FieldVariableName	string	–
{of}	{of | species}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
Value_type	string	–
{=}	{= | are | is}	–
Angular_velocity	real. . .	–

Ic Couette_X

Syntax

Ic Couette_X *{on} MeshPartIdentifier FieldVariableName* ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*]) *Coef {=} Parameters. . .*

Summary

Spatially dependent circular x velocity initial condition valid in an Aria region. IC depends upon its coordinate.

Parameter	Value	Default
{on}	{@ | at | for | in | on | over}	–
MeshPartIdentifier	string	–
FieldVariableName	string	–
{of}	{of | species}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
Coef	string	–
{=}	{= | are | is}	–
Parameters	real. . .	–

Ic Couette_Y

Syntax

Ic Couette_Y *{on} MeshPartIdentifier FieldVariableName* ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*]) *Coef {=} Parameters. . .*

Summary

Spatially dependent circular y velocity initial condition valid in an Aria region. IC depends upon its coordinate.

Parameter	Value	Default
{on}	{@ | at | for | in | on | over}	–
MeshPartIdentifier	string	–
FieldVariableName	string	–
{of}	{of | species}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
Coef	string	–
{=}	{= | are | is}	–
Parameters	real. . .	–

Ic Couette_Sh

Syntax

Ic Couette_Sh *{on} MeshPartIdentifier FieldVariableName* ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*]) *Coef {=} Parameters. . .*

Summary

Spatially dependent initial condition for shear rate valid in an Aria region. IC depends upon its coordinate.

Parameter	Value	Default
{on}	{@ | at | for | in | on | over}	–
MeshPartIdentifier	string	–
FieldVariableName	string	–
{of}	{of | species}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
Coef	string	–
{=}	{= | are | is}	–
Parameters	real. . .	–

Ic Tangent

Syntax

Ic Tangent *FieldVariableName* ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*]) *{on} SurfaceName*

Summary

The given field named FieldVariableName is to be made tangent to the surface named SurfaceName.

Parameter	Value	Default
FieldVariableName	string	–
{of}	{of | species}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{on}	{@ | at | for | in | on | over}	–
SurfaceName	string	–

7.17 Sources

Equations

A list of equations for which sources can be applied is provided below

Ampere (

Constant_Vector

Syntax

Source For Ampere ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Constant_Vector ([X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
x	real	0
y	real	0
z	real	0

Grad_Voltage

Syntax

Source For Ampere ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Grad_Voltage ()

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–

User_Field

Syntax

Source For Ampere ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a Vector User Field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Source For Ampere ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = User_Function ([Name_X = *name_x*] | [Name_Y = *name_y*] | [Name_Z = *name_z*] | X = *x* | [Multiplier = *multiplier*])

Summary

Value from a Vector User Function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name_x	"string"	–
name_y	"string"	–
name_z	"string"	–
x	"string"	–
multiplier	real	1

Vector_String_Function

Syntax

Source For Ampere ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Vector_String_Function (F_X = *f_x* | F_Y = *f_y* | [F_Z = *f_z*])

Summary

Value from a string function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
f_x	"string"	–
f_y	"string"	–
f_z	"string"	–

Bar Momentum (

Constant

Syntax

Source For Bar Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Constant Value = *value*

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
value	real	–

Exponential

Syntax

Source For Bar Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The source, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Source For Bar Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Global (Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
global_name	"string"	–
component	integer	0

Gravity

Syntax

Source For Bar Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Gravity (F_X = *f_x* | F_Y = *f_y* | [F_Z = *f_z*])

Summary

Gravitation body force for advective bar pressure solve

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
f_x	"string"	–
f_y	"string"	–
f_z	"string"	0.0

Lumped_Constant

Syntax

Source For Bar Momentum ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = Lumped_Constant Value = *value*

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
value	real	–

Lumped_Exponential

Syntax

Source For Bar Momentum ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = Lumped_Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Lumped_Global

Syntax

Source For Bar Momentum ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = Lumped_Global (Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
global_name	"string"	–
component	integer	0

Lumped_Nd_Table

Syntax

Source For Bar Momentum ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = Lumped_Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Lumped_Polynomial

Syntax

Source For Bar Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Lumped_Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Lumped_Scalar_String_Function

Syntax

Source For Bar Momentum ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = Lumped_Scalar_String_Function F = *f*

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
f	"string"	–

Lumped_User_Field

Syntax

Source For Bar Momentum ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = Lumped_User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

Lumped_User_Function

Syntax

Source For Bar Momentum ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = Lumped_User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1
toggle	"string"	–

Major_Loss

Syntax

Source For Bar Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Major_Loss [Scale = *scale*]

Summary

Major friction loss term for 1D advective bar pressure solve

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
scale	real	0.125

Nd_Table

Syntax

Source For Bar Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Source For Bar Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The source, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Source For Bar Momentum ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = Scalar_String_Function F = *f*

Summary

Value from a string function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
f	"string"	–

User_Field

Syntax

Source For Bar Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name	“string”	–
scaling	real	1
global_var	“string”	–
time_function	“string”	–

User_Function

Syntax

Source For Bar Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1
toggle	"string"	–

User_Plugin

Syntax

Source For Bar Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = User_Plugin (Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
Name	"string"	–
plugin_parameters	"string"	–

Body Acceleration (

Constant_Vector

Syntax

Source For Body Acceleration ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Constant_Vector ([Valuex = *valuex*] | [Valuey = *valuey*] | [Valuez = *valuez*])

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
valuex	real	0
valuey	real	0
valuez	real	0

User_Field

Syntax

Source For Body Acceleration ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a Vector User Field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name	“string”	–
scaling	real	1
global_var	“string”	–
time_function	“string”	–

User_Function

Syntax

Source For Body Acceleration ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = User_Function ([Name_X = *name_x*] | [Name_Y = *name_y*] | [Name_Z = *name_z*] | X = *x* | [Multiplier = *multiplier*])

Summary

Value from a Vector User Function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name_x	“string”	–
name_y	“string”	–
name_z	“string”	–
x	“string”	–
multiplier	real	1

Vector_String_Function

Syntax

Source For Body Acceleration ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = Vector_String_Function (F_X = *f_x* | F_Y = *f_y* | [F_Z = *f_z*])

Summary

Value from a string function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
f_x	"string"	–
f_y	"string"	–
f_z	"string"	–

Brinkman Momentum (

Constant_Vector

Syntax

Source For Brinkman Momentum ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = Constant_Vector ([Valuex = *valuex*] | [Valuey = *valuey*] | [Valuez = *valuez*])

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
valuex	real	0
valuey	real	0
valuez	real	0

User_Field

Syntax

Source For Brinkman Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a Vector User Field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name	“string”	–
scaling	real	1
global_var	“string”	–
time_function	“string”	–

User_Function

Syntax

Source For Brinkman Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = User_Function ([Name_X = *name_x*] | [Name_Y = *name_y*] | [Name_Z = *name_z*] | X = *x* | [Multiplier = *multiplier*])

Summary

Value from a Vector User Function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name_x	"string"	–
name_y	"string"	–
name_z	"string"	–
x	"string"	–
multiplier	real	1

Vector_String_Function

Syntax

Source For Brinkman Momentum ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = Vector_String_Function (F_X = *f_x* | F_Y = *f_y* | [F_Z = *f_z*])

Summary

Value from a string function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
f_x	"string"	–
f_y	"string"	–
f_z	"string"	–

Charge Density (

Constant

Syntax

Source For Charge Density ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Constant Value = *value*

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
value	real	–

Exponential

Syntax

Source For Charge Density ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The source, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Source For Charge Density ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = Global (Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
global_name	"string"	–
component	integer	0

Lumped_Constant

Syntax

Source For Charge Density ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = Lumped_Constant Value = *value*

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
value	real	–

Lumped_Exponential

Syntax

Source For Charge Density ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Lumped_Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
variable	“string”	–
constant	real	0
multiplier	real	1
exponent	real	–

Lumped_Global

Syntax

Source For Charge Density ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Lumped_Global (Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
global_name	“string”	–
component	integer	0

Lumped_Nd_Table

Syntax

Source For Charge Density ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Lumped_Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Lumped_Polynomial

Syntax

Source For Charge Density ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Lumped_Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
variable	“string”	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Lumped_Scalar_String_Function

Syntax

Source For Charge Density ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Lumped_Scalar_String_Function F = *f*

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be

a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
f	"string"	–

Lumped_User_Field

Syntax

Source For Charge Density ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Lumped_User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

Lumped_User_Function

Syntax

Source For Charge Density ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Lumped_User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

Nd_Table

Syntax

Source For Charge Density ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Source For Charge Density ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The source, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Source For Charge Density ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Scalar_String_Function F = *f*

Summary

Value from a string function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
f	"string"	–

User_Field

Syntax

Source For Charge Density ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name	“string”	–
scaling	real	1
global_var	“string”	–
time_function	“string”	–

User_Function

Syntax

Source For Charge Density ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1
toggle	"string"	–

User_Plugin

Syntax

Source For Charge Density ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = User_Plugin (Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
Name	"string"	–
plugin_parameters	"string"	–

Co2 Mass Balance (

Constant

Syntax

Source For Co2 Mass Balance ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Constant Value = *value*

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
value	real	–

Exponential

Syntax

Source For Co2 Mass Balance ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The source, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
variable	“string”	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Source For Co2 Mass Balance ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Global (Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
global_name	“string”	–
component	integer	0

Lumped_Constant

Syntax

Source For Co2 Mass Balance ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Lumped_Constant Value = *value*

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
value	real	–

Lumped_Exponential

Syntax

Source For Co2 Mass Balance ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Lumped_Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Lumped_Global

Syntax

Source For Co2 Mass Balance ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Lumped_Global (Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
global_name	"string"	–
component	integer	0

Lumped_Nd_Table

Syntax

Source For Co2 Mass Balance ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Lumped_Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Lumped_Polynomial

Syntax

Source For Co2 Mass Balance ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Lumped_Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Lumped_Scalar_String_Function

Syntax

Source For Co2 Mass Balance ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = Lumped_Scalar_String_Function F = *f*

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
f	"string"	–

Lumped_User_Field

Syntax

Source For Co2 Mass Balance ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Lumped_User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

Lumped_User_Function

Syntax

Source For Co2 Mass Balance ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Lumped_User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

Nd_Table

Syntax

Source For Co2 Mass Balance ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Source For Co2 Mass Balance ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The source, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Source For Co2 Mass Balance ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = Scalar_String_Function F = *f*

Summary

Value from a string function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
f	"string"	–

User_Field

Syntax

Source For Co2 Mass Balance ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name	“string”	–
scaling	real	1
global_var	“string”	–
time_function	“string”	–

User_Function

Syntax

Source For Co2 Mass Balance ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1
toggle	"string"	–

User_Plugin

Syntax

Source For Co2 Mass Balance ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = User_Plugin (Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
Name	"string"	–
plugin_parameters	"string"	–

Continuity (

Constant

Syntax

Source For Continuity ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = Constant Value = *value*

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
value	real	–

Exponential

Syntax

Source For Continuity ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The source, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
variable	“string”	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Source For Continuity ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Global (Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
global_name	“string”	–
component	integer	0

Lumped_Constant

Syntax

Source For Continuity ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Lumped_Constant Value = *value*

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
value	real	–

Lumped_Exponential

Syntax

Source For Continuity ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Lumped_Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Lumped_Global

Syntax

Source For Continuity ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Lumped_Global (Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
global_name	"string"	–
component	integer	0

Lumped_Nd_Table

Syntax

Source For Continuity ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Lumped_Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Lumped_Polynomial

Syntax

Source For Continuity ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Lumped_Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Lumped_Scalar_String_Function

Syntax

Source For Continuity ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Lumped_Scalar_String_Function F = *f*

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
f	"string"	–

Lumped_User_Field

Syntax

Source For Continuity ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Lumped_User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

Lumped_User_Function

Syntax

Source For Continuity ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Lumped_User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1
toggle	"string"	–

Mixed_Elasticity

Syntax

Source For Continuity ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = Mixed_Elasticity ()

Summary

Mixed elasticity source for continuity equation

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–

Nd_Table

Syntax

Source For Continuity ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Source For Continuity ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The source, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Source For Continuity ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = Scalar_String_Function F = *f*

Summary

Value from a string function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
f	"string"	–

User_Field

Syntax

Source For Continuity ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Source For Continuity ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Source For Continuity ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = User_Plugin (Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
Name	“string”	–
plugin_parameters	“string”	–

Current (

Butler_Volmer

Syntax

Source For Current ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Butler_Volmer ([R = *r*] | [F = *f*] | [Alpha_A = *alpha_a*] | [Alpha_C = *alpha_c*] | [Flux_Lim_A = *flux_lim_a*]

| [Flux_Lim_C = *flux_lim_c*] | [Lumped = *lumped*] | [Sign = *sign*] | [A = *a*] | [K = *k*] | [Species_A = *species_a*] | [Species_B = *species_b*])

Summary

Butler volmer source for current equation to be used with porous electrodes

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
r	real	8.314
f	real	9.649e+04
alpha_a	real	0.5
alpha_c	real	0.5
flux_lim_a	real	0
flux_lim_c	real	0
lumped	integer	–
sign	real	1
a	real	0
k	real	0
species_a	“string”	–
species_b	“string”	–

Butler_Volmer_From_Liquid_Phase

Syntax

Source For Current ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* =
Butler_Volmer_From_Liquid_Phase Mult = *Mult*

Summary

Butler volmer source for current equation to be used with porous electrodes

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
Mult	real	–

Butler_Volmer_Linear

Syntax

Source For Current ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = Butler_Volmer_Linear (Species_A = *species_a* | Species_B = *species_b* | A = *a* | U = *u* | R = *r* | F = *f* | K = *k*)

Summary

This model has been deprecated and removed from the code.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
species_a	“string”	–
species_b	“string”	–
a	real	–
u	real	–
r	real	–
f	real	–
k	real	–

Butler_Volmer_Simple

Syntax

Source For Current ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = Butler_Volmer_Simple ([V1_Subindex = *v1_subindex*] | [V2_Subindex = *v2_subindex*] | A = *a* | C_A = *c_a* | C_C = *c_c* | U = *u* | Sign = *sign*)

Summary

Butler volmer simple source for current equation

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
v1_subindex	integer	1
v2_subindex	integer	2
a	real	–
c_a	real	–
c_c	real	–
u	real	–
sign	real	–

Constant

Syntax

Source For Current ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = Constant Value = *value*

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
value	real	–

Electrode_Object

Syntax

Source For Current ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Electrode_Object ()

Summary

Electrode object source for current equation to be used with porous electrodes

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–

Electrode_Object_Old

Syntax

Source For Current ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Electrode_Object_Old (F = *f* | Electrodefile = *electrodeFile* | Kmolconversion = *kmolConversion* | [Jconversion = *JConversion*] | [Meterconversion = *meterConversion*] | [Max_Sub_Timesteps = *max_sub_timesteps*] | [Base_Delta = *base_delta*])

Summary

Electrode object source for current equation to be used with porous electrodes

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
f	real	–
electrodeFile	“string”	–
kmolConversion	real	–
JConversion	real	1
meterConversion	real	1
max_sub_timesteps	integer	100
base_delta	real	1e-05

Exponential

Syntax

Source For Current ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The source, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
variable	“string”	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Source For Current ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Global (Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
global_name	"string"	–
component	integer	0

Lumped_Constant

Syntax

Source For Current ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = Lumped_Constant Value = *value*

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
value	real	–

Lumped_Electrode_Object

Syntax

Source For Current ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = Lumped_Electrode_Object ()

Summary

Electrode object source for current equation to be used with porous electrodes

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–

Lumped_Exponential

Syntax

Source For Current ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Lumped_Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
variable	“string”	–
constant	real	0
multiplier	real	1
exponent	real	–

Lumped_Global

Syntax

Source For Current ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Lumped_Global (Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
global_name	“string”	–
component	integer	0

Lumped_Nd_Table

Syntax

Source For Current ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Lumped_Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Lumped_Polynomial

Syntax

Source For Current ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Lumped_Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Lumped_Scalar_String_Function

Syntax

Source For Current ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Lumped_Scalar_String_Function F = *f*

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be

a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
f	"string"	–

Lumped_User_Field

Syntax

Source For Current ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Lumped_User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

Lumped_User_Function

Syntax

Source For Current ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Lumped_User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

Nd_Table

Syntax

Source For Current ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Source For Current ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The source, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Source For Current ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Scalar_String_Function F = *f*

Summary

Value from a string function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
f	"string"	–

User_Field

Syntax

Source For Current ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Source For Current ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Source For Current ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = User_Plugin (Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
Name	“string”	–
plugin_parameters	“string”	–

Cvfem Continuity (

Constant

Syntax

Source For Cvfem Continuity ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Constant Value = *value*

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
value	real	–

Exponential

Syntax

Source For Cvfem Continuity ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The source, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
variable	“string”	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Source For Cvfem Continuity ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Global (Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
global_name	“string”	–
component	integer	0

Lumped_Constant

Syntax

Source For Cvfem Continuity ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Lumped_Constant Value = *value*

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
value	real	–

Lumped_Exponential

Syntax

Source For Cvfem Continuity ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Lumped_Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Lumped_Global

Syntax

Source For Cvfem Continuity ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Lumped_Global (Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
global_name	"string"	–
component	integer	0

Lumped_Nd_Table

Syntax

Source For Cvfem Continuity ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Lumped_Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Lumped_Polynomial

Syntax

Source For Cvfem Continuity ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Lumped_Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Lumped_Scalar_String_Function

Syntax

Source For Cvfem Continuity ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = Lumped_Scalar_String_Function F = *f*

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
f	"string"	–

Lumped_User_Field

Syntax

Source For Cvfem Continuity ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = Lumped_User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

Lumped_User_Function

Syntax

Source For Cvfem Continuity ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = Lumped_User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1
toggle	"string"	–

Moving_Mesh

Syntax

Source For Cvfem Continuity ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Moving_Mesh ()

Summary

Moving mesh source for conservative form, continuity. Switches between forms based on activate vrtm flag.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–

Nd_Table

Syntax

Source For Cvfem Continuity ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Source For Cvfem Continuity ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The source, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Source For Cvfem Continuity ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = Scalar_String_Function F = *f*

Summary

Value from a string function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
f	"string"	–

User_Field

Syntax

Source For Cvfem Continuity ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name	“string”	–
scaling	real	1
global_var	“string”	–
time_function	“string”	–

User_Function

Syntax

Source For Cvfem Continuity ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1
toggle	"string"	–

User_Plugin

Syntax

Source For Cvfem Continuity ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = User_Plugin (Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
Name	"string"	–
plugin_parameters	"string"	–

Cvfem Energy (

Cantera_Homogeneous_Chemistry

Syntax

Source For Cvfem Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Cantera_Homogeneous_Chemistry Mechanism = *mechanism*

Summary

Cantera energy source for CVFEM energy equation

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
mechanism	"string"	–

Chemistry

Syntax

Source For Cvfem Energy ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = Chemistry Mechanism = *mechanism*

Summary

Chemical reaction source for energy equation

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
mechanism	"string"	–

Constant

Syntax

Source For Cvfem Energy ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = Constant Value = *value*

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
value	real	–

Exponential

Syntax

Source For Cvfem Energy ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The source, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
variable	“string”	–
constant	real	0
multiplier	real	1
exponent	real	–

Full_Viscous_Dissipation

Syntax

Source For Cvfem Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Full_Viscous_Dissipation [Multiplier = *multiplier*]

Summary

CVFEM full viscous dissipation source for cvfem energy equation

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
multiplier	real	1

Global

Syntax

Source For Cvfem Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Global (Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
global_name	“string”	–
component	integer	0

Lumped_Constant

Syntax

Source For Cvfem Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Lumped_Constant Value = *value*

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
value	real	–

Lumped_Exponential

Syntax

Source For Cvfem Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Lumped_Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
variable	“string”	–
constant	real	0
multiplier	real	1
exponent	real	–

Lumped_Global

Syntax

Source For Cvfem Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Lumped_Global (Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
global_name	“string”	–
component	integer	0

Lumped_Nd_Table

Syntax

Source For Cvfem Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Lumped_Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Lumped_Polynomial

Syntax

Source For Cvfem Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Lumped_Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Lumped_Scalar_String_Function

Syntax

Source For Cvfem Energy ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = Lumped_Scalar_String_Function F = *f*

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
f	"string"	–

Lumped_User_Field

Syntax

Source For Cvfem Energy ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = Lumped_User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

Lumped_User_Function

Syntax

Source For Cvfem Energy ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = Lumped_User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1
toggle	"string"	–

Melting

Syntax

Source For Cvfem Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Melting ([Ts = *Ts*] | [Tl = *Tl*] | [Latent_Heat = *latent_heat*] | [Profile_Shape = *profile_shape*] | [Add_Advection = *add_advection*])

Summary

Energy source term to represent the latent heat of melting of a material.

Description

Uses a Gaussian (default) or Square distribution based on solidus (TS) and liquidus (TL) temperatures to compute a liquid fraction f_L. The Gaussian option is set so that only one percent of the energy gets applied outside of this range. The source term is computed as:

$$\rho L \frac{f_L^{n+1} - f_L{}^n}{\Delta t}$$

Uses either the LATENT_HEAT (L) expression or a provided "latent_heat = X" argument for each source. If both are present, the provided argument is used. This also assumes that the LATENT_HEAT is provided per mass (this expression multiplies it by the density). The solidus and liquidus temperatures

can either be provided inline or in the material model. If both are provided, the inline values are used. The ADD_ADVECTION option adds an additional term that tracks the advection of the solidification front, which is currently implemented as:

$$\rho L \frac{\partial f_L}{\partial T} \boldsymbol{u} \cdot \nabla T$$

If this option is enabled, users should be aware that this term may be subject to temporal errors where large enough time steps can advance the solidification front over and incorrectly withhold energy over many cells.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
Ts	real	0
Tl	real	0
latent_heat	real	0
profile_shape	"string"	gaussian
add_advection	integer	–

Moving_Mesh

Syntax

Source For Cvfem Energy ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = Moving_Mesh ()

Summary

Moving mesh source for conservative form, energy. Switches between forms based on activate vrtm flag.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–

Nd_Table

Syntax

Source For Cvfem Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Source For Cvfem Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The source, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Pressure_Material_Derivative

Syntax

Source For Cvfem Energy ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = Pressure_Material_Derivative [Multiplier = *multiplier*]

Summary

CVFEM pressure material derivative source for cvfem energy equation

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
multiplier	real	1

Scalar_String_Function

Syntax

Source For Cvfem Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Scalar_String_Function F = *f*

Summary

Value from a string function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
f	"string"	–

User_Field

Syntax

Source For Cvfem Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Source For Cvfem Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Source For Cvfem Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = User_Plugin (Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
Name	"string"	–
plugin_parameters	"string"	–

Viscous_Dissipation

Syntax

Source For Cvfem Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Viscous_Dissipation [Multiplier = *multiplier*]

Summary

CVFEM viscous dissipation source for cvfem energy equation

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
multiplier	real	1

Cvfem Level Set (

Constant

Syntax

Source For Cvfem Level Set ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Constant Value = *value*

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
value	real	–

Cvfem_Ls_Classic

Syntax

Source For Cvfem Level Set ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Cvfem_Ls_Classic ([Crank_Nicolson = *crank_nicolson*] | [Half_Velocity = *half_velocity*])

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
crank_nicolson	integer	–
half_velocity	integer	–

Exponential

Syntax

Source For Cvfem Level Set ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The source, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Source For Cvfem Level Set ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Global (Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
global_name	"string"	–
component	integer	0

Lumped_Constant

Syntax

Source For Cvfem Level Set ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Lumped_Constant Value = *value*

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
value	real	–

Lumped_Exponential

Syntax

Source For Cvfem Level Set ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = Lumped_Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
variable	“string”	–
constant	real	0
multiplier	real	1
exponent	real	–

Lumped_Global

Syntax

Source For Cvfem Level Set ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = Lumped_Global (Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
global_name	“string”	–
component	integer	0

Lumped_Nd_Table

Syntax

Source For Cvfem Level Set ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Lumped_Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Lumped_Polynomial

Syntax

Source For Cvfem Level Set ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Lumped_Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
variable	“string”	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Lumped_Scalar_String_Function

Syntax

Source For Cvfem Level Set ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Lumped_Scalar_String_Function F = *f*

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be

a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
f	"string"	–

Lumped_User_Field

Syntax

Source For Cvfem Level Set ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Lumped_User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

Lumped_User_Function

Syntax

Source For Cvfem Level Set ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Lumped_User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

Nd_Table

Syntax

Source For Cvfem Level Set ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Source For Cvfem Level Set ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The source, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Source For Cvfem Level Set ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = Scalar_String_Function F = *f*

Summary

Value from a string function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
f	"string"	–

User_Field

Syntax

Source For Cvfem Level Set ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name	“string”	–
scaling	real	1
global_var	“string”	–
time_function	“string”	–

User_Function

Syntax

Source For Cvfem Level Set ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1
toggle	"string"	–

User_Plugin

Syntax

Source For Cvfem Level Set ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = User_Plugin (Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
Name	"string"	–
plugin_parameters	"string"	–

Cvfem Mass Fraction (

Cantera_Homogeneous_Chemistry

Syntax

Source For Cvfem Mass Fraction ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over}*

MeshPartIdentifier = Cantera_Homogeneous_Chemistry Mechanism = *mechanism*

Summary

Cantera mass fraction source for CVFEM mass fraction equation

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
mechanism	"string"	–

Constant

Syntax

Source For Cvfem Mass Fraction ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = Constant Value = *value*

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
value	real	–

Edc_Mass_Fraction

Syntax

Source For Cvfem Mass Fraction ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = Edc_Mass_Fraction ()

Summary

EDC mass fraction source for CVFEM mass fraction equation

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–

Exponential

Syntax

Source For Cvfem Mass Fraction ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The source, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
variable	“string”	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Source For Cvfem Mass Fraction ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = Global (Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
global_name	“string”	–
component	integer	0

Lumped_Constant

Syntax

Source For Cvfem Mass Fraction ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = Lumped_Constant Value = *value*

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
value	real	–

Lumped_Exponential

Syntax

Source For Cvfem Mass Fraction ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Lumped_Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Lumped_Global

Syntax

Source For Cvfem Mass Fraction ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Lumped_Global (Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
global_name	"string"	–
component	integer	0

Lumped_Nd_Table

Syntax

Source For Cvfem Mass Fraction ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Lumped_Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Lumped_Polynomial

Syntax

Source For Cvfem Mass Fraction ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Lumped_Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Lumped_Scalar_String_Function

Syntax

Source For Cvfem Mass Fraction ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Lumped_Scalar_String_Function F = *f*

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
f	“string”	–

Lumped_User_Field

Syntax

Source For Cvfem Mass Fraction ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = Lumped_User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name	“string”	–
scaling	real	1
global_var	“string”	–
time_function	“string”	–

Lumped_User_Function

Syntax

Source For Cvfem Mass Fraction ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = Lumped_User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1
toggle	"string"	–

Moving_Mesh

Syntax

Source For Cvfem Mass Fraction ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = Moving_Mesh ()

Summary

Moving mesh source for conservative form, mass fraction. Switches between forms based on activate vrtm flag.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–

Nd_Table

Syntax

Source For Cvfem Mass Fraction ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Source For Cvfem Mass Fraction ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The source, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Source For Cvfem Mass Fraction ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = Scalar_String_Function F = *f*

Summary

Value from a string function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
f	"string"	–

User_Field

Syntax

Source For Cvfem Mass Fraction ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name	“string”	–
scaling	real	1
global_var	“string”	–
time_function	“string”	–

User_Function

Syntax

Source For Cvfem Mass Fraction ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Source For Cvfem Mass Fraction ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = User_Plugin (Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
Name	“string”	–
plugin_parameters	“string”	–

Cvfem Mesh (

Constant_Vector

Syntax

Source For Cvfem Mesh ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = Constant_Vector ([Valuex = *valuex*] | [Valuey = *valuey*] | [Valuez = *valuez*])

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
valuex	real	0
valuey	real	0
valuez	real	0

Mesh_Body_Acceleration

Syntax

Source For Cvfem Mesh ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = Mesh_Body_Acceleration ()

Summary

mesh body acceleration source for mesh equation

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–

User_Field

Syntax

Source For Cvfem Mesh ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a Vector User Field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Source For Cvfem Mesh ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = User_Function ([Name_X = *name_x*] | [Name_Y = *name_y*] | [Name_Z = *name_z*] | X = *x* | [Multiplier = *multiplier*])

Summary

Value from a Vector User Function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name_x	"string"	–
name_y	"string"	–
name_z	"string"	–
x	"string"	–
multiplier	real	1

Vector_String_Function

Syntax

Source For Cvfem Mesh ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Vector_String_Function (F_X = *f_x* | F_Y = *f_y* | [F_Z = *f_z*])

Summary

Value from a string function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
f_x	“string”	–
f_y	“string”	–
f_z	“string”	–

Cvfem Mixture Fraction (

Constant

Syntax

Source For Cvfem Mixture Fraction ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Constant Value = *value*

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
value	real	–

Exponential

Syntax

Source For Cvfem Mixture Fraction ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The source, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
variable	“string”	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Source For Cvfem Mixture Fraction ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Global (Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
global_name	"string"	–
component	integer	0

Lumped_Constant

Syntax

Source For Cvfem Mixture Fraction ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = Lumped_Constant Value = *value*

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
value	real	–

Lumped_Exponential

Syntax

Source For Cvfem Mixture Fraction ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = Lumped_Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Lumped_Global

Syntax

Source For Cvfem Mixture Fraction ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = Lumped_Global (Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
global_name	"string"	–
component	integer	0

Lumped_Nd_Table

Syntax

Source For Cvfem Mixture Fraction ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = Lumped_Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Lumped_Polynomial

Syntax

Source For Cvfem Mixture Fraction ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Lumped_Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Lumped_Scalar_String_Function

Syntax

Source For Cvfem Mixture Fraction ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = Lumped_Scalar_String_Function F = *f*

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
f	"string"	–

Lumped_User_Field

Syntax

Source For Cvfem Mixture Fraction ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = Lumped_User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

Lumped_User_Function

Syntax

Source For Cvfem Mixture Fraction ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = Lumped_User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1
toggle	"string"	–

Nd_Table

Syntax

Source For Cvfem Mixture Fraction ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Source For Cvfem Mixture Fraction ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The source, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Source For Cvfem Mixture Fraction ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = Scalar_String_Function F = *f*

Summary

Value from a string function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
f	"string"	–

User_Field

Syntax

Source For Cvfem Mixture Fraction ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name	“string”	–
scaling	real	1
global_var	“string”	–
time_function	“string”	–

User_Function

Syntax

Source For Cvfem Mixture Fraction ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Source For Cvfem Mixture Fraction ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = User_Plugin (Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
Name	“string”	–
plugin_parameters	“string”	–

Cvfem Momentum (

Boussinesq

Syntax

Source For Cvfem Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Boussinesq (Ref_Temperature = *ref_temperature* | [Gx = *gx*] | [Gy = *gy*] | [Gz = *gz*])

Summary

CVFEM Boussinesq source for cvfem momentum equation

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
ref_temperature	real	–
gx	real	0
gy	real	0
gz	real	0

Constant_Vector

Syntax

Source For Cvfem Momentum ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = Constant_Vector ([Valuex = *valuex*] | [Valuey = *valuey*] | [Valuez = *valuez*])

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
valuex	real	0
valuey	real	0
valuez	real	0

Hydrostatic

Syntax

Source For Cvfem Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Hydrostatic ([Gx = *gx*] | [Gy = *gy*] | [Gz = *gz*] | [Ref_Density = *ref_density*])

Summary

CVFEM Hydrostatic source for cvfem momentum equation

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
gx	real	0
gy	real	0
gz	real	0
ref_density	real	0

Ls_Heaviside_Capillary

Syntax

Source For Cvfem Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Ls_Heaviside_Capillary ()

Summary

LS heaviside capillary source for momentum equation

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–

Moving_Mesh

Syntax

Source For Cvfem Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Moving_Mesh ()

Summary

Moving mesh source for conservative form. Switches between forms based on activate vrtm flag.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–

Relative_Darcy

Syntax

Source For Cvfem Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Relative_Darcy Relative_To = *relative_to*

Summary

CVFEM Relative Darcy source for cvfem momentum equation

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
relative_to	“string”	–

Static_Porous_Term

Syntax

Source For Cvfem Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Static_Porous_Term ()

Summary

Source Term required when material flow is coupled with porous fluid flow. The term is needed when the gas phase pressure can effect the motion of the matrix phase With out this term, at a constant pressure with a porosity gradient there will be motion of the matrix phase. $\vec{s} = -P_g \nabla \phi$

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–

User_Field

Syntax

Source For Cvfem Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a Vector User Field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name	“string”	–
scaling	real	1
global_var	“string”	–
time_function	“string”	–

User_Function

Syntax

Source For Cvfem Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = User_Function ([Name_X = *name_x*] | [Name_Y = *name_y*] | [Name_Z = *name_z*] | X = *x* | [Multiplier = *multiplier*])

Summary

Value from a Vector User Function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name_x	“string”	–
name_y	“string”	–
name_z	“string”	–
x	“string”	–
multiplier	real	1

Vector_String_Function

Syntax

Source For Cvfem Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Vector_String_Function (F_X = *f_x* | F_Y = *f_y* | [F_Z = *f_z*])

Summary

Value from a string function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
f_x	"string"	–
f_y	"string"	–
f_z	"string"	–

Cvfem Specific Dissipation Rate (

Constant

Syntax

Source For Cvfem Specific Dissipation Rate ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Constant Value = *value*

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
value	real	–

Exponential

Syntax

Source For Cvfem Specific Dissipation Rate ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The source, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Source For Cvfem Specific Dissipation Rate ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = Global (Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
global_name	"string"	–
component	integer	0

Komega_Sdr

Syntax

Source For Cvfem Specific Dissipation Rate ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Komega_Sdr ()

Summary

K-omega source for CVFEM specific dissipation equation

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–

Lumped_Constant

Syntax

Source For Cvfem Specific Dissipation Rate ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Lumped_Constant Value = *value*

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
value	real	–

Lumped_Exponential

Syntax

Source For Cvfem Specific Dissipation Rate ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = Lumped_Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Lumped_Global

Syntax

Source For Cvfem Specific Dissipation Rate ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = Lumped_Global (Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
global_name	"string"	–
component	integer	0

Lumped_Nd_Table

Syntax

Source For Cvfem Specific Dissipation Rate ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = Lumped_Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Lumped_Polynomial

Syntax

Source For Cvfem Specific Dissipation Rate ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = Lumped_Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Lumped_Scalar_String_Function

Syntax

Source For Cvfem Specific Dissipation Rate ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = Lumped_Scalar_String_Function F = *f*

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
f	"string"	–

Lumped_User_Field

Syntax

Source For Cvfem Specific Dissipation Rate ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = Lumped_User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

Lumped_User_Function

Syntax

Source For Cvfem Specific Dissipation Rate ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = Lumped_User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1
toggle	"string"	–

Nd_Table

Syntax

Source For Cvfem Specific Dissipation Rate ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Source For Cvfem Specific Dissipation Rate ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The source, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Source For Cvfem Specific Dissipation Rate ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = Scalar_String_Function F = *f*

Summary

Value from a string function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
f	"string"	–

Sst_Sdr

Syntax

Source For Cvfem Specific Dissipation Rate ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Sst_Sdr ()

Summary

K-omega SST source for specific dissipation equation

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–

User_Field

Syntax

Source For Cvfem Specific Dissipation Rate ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name	“string”	–
scaling	real	1
global_var	“string”	–
time_function	“string”	–

User_Function

Syntax

Source For Cvfem Specific Dissipation Rate ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Source For Cvfem Specific Dissipation Rate ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = User_Plugin (Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
Name	"string"	–
plugin_parameters	"string"	–

Cvfem Turbulence Dissipation Rate (

Constant

Syntax

Source For Cvfem Turbulence Dissipation Rate ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = Constant Value = *value*

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
value	real	–

Exponential

Syntax

Source For Cvfem Turbulence Dissipation Rate ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The source, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Source For Cvfem Turbulence Dissipation Rate ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = Global (Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
global_name	"string"	–
component	integer	0

Kepsilon_Turb_Dr

Syntax

Source For Cvfem Turbulence Dissipation Rate ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = Kepsilon_Turb_Dr ()

Summary

K-epsilon source for CVFEM turbulence dissipation equation

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–

Lumped_Constant

Syntax

Source For Cvfem Turbulence Dissipation Rate ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = Lumped_Constant Value = *value*

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
value	real	–

Lumped_Exponential

Syntax

Source For Cvfem Turbulence Dissipation Rate ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Lumped_Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Lumped_Global

Syntax

Source For Cvfem Turbulence Dissipation Rate ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Lumped_Global (Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
global_name	"string"	–
component	integer	0

Lumped_Nd_Table

Syntax

Source For Cvfem Turbulence Dissipation Rate ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Lumped_Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Lumped_Polynomial

Syntax

Source For Cvfem Turbulence Dissipation Rate ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Lumped_Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Lumped_Scalar_String_Function

Syntax

Source For Cvfem Turbulence Dissipation Rate ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = Lumped_Scalar_String_Function F = *f*

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
f	“string”	–

Lumped_User_Field

Syntax

Source For Cvfem Turbulence Dissipation Rate ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = Lumped_User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name	“string”	–
scaling	real	1
global_var	“string”	–
time_function	“string”	–

Lumped_User_Function

Syntax

Source For Cvfem Turbulence Dissipation Rate ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = Lumped_User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1
toggle	"string"	–

Nd_Table

Syntax

Source For Cvfem Turbulence Dissipation Rate ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Source For Cvfem Turbulence Dissipation Rate ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The source, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Source For Cvfem Turbulence Dissipation Rate ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = Scalar_String_Function F = *f*

Summary

Value from a string function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
f	"string"	–

User_Field

Syntax

Source For Cvfem Turbulence Dissipation Rate ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name	“string”	–
scaling	real	1
global_var	“string”	–
time_function	“string”	–

User_Function

Syntax

Source For Cvfem Turbulence Dissipation Rate ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1
toggle	"string"	–

User_Plugin

Syntax

Source For Cvfem Turbulence Dissipation Rate ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = User_Plugin (Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
Name	"string"	–
plugin_parameters	"string"	–

Cvfem Turbulent Kinetic Energy (

Constant

Syntax

Source For Cvfem Turbulent Kinetic Energy ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = Constant Value = *value*

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
value	real	–

Exponential

Syntax

Source For Cvfem Turbulent Kinetic Energy ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The source, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
variable	“string”	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Source For Cvfem Turbulent Kinetic Energy ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = Global (Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
global_name	“string”	–
component	integer	0

Kepsilon_Turb_Ke

Syntax

Source For Cvfem Turbulent Kinetic Energy ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = Kepsilon_Turb_Ke ()

Summary

K-epsilon turbulent source for CVFEM turbulent kinetic energy equation

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–

Komega_Turb_Ke

Syntax

Source For Cvfem Turbulent Kinetic Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Komega_Turb_Ke ()

Summary

K-omega turbulent source for CVFEM turbulent kinetic energy equation

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–

Ksgs_Turb_Ke

Syntax

Source For Cvfem Turbulent Kinetic Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Ksgs_Turb_Ke ()

Summary

KSGS turbulent source for CVFEM turbulent kinetic energy equation

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–

Lumped_Constant

Syntax

Source For Cvfem Turbulent Kinetic Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Lumped_Constant Value = *value*

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
value	real	–

Lumped_Exponential

Syntax

Source For Cvfem Turbulent Kinetic Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Lumped_Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
variable	“string”	–
constant	real	0
multiplier	real	1
exponent	real	–

Lumped_Global

Syntax

Source For Cvfem Turbulent Kinetic Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Lumped_Global (Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
global_name	“string”	–
component	integer	0

Lumped_Nd_Table

Syntax

Source For Cvfem Turbulent Kinetic Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Lumped_Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Lumped_Polynomial

Syntax

Source For Cvfem Turbulent Kinetic Energy ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = Lumped_Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Lumped_Scalar_String_Function

Syntax

Source For Cvfem Turbulent Kinetic Energy ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = Lumped_Scalar_String_Function F = *f*

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
f	"string"	–

Lumped_User_Field

Syntax

Source For Cvfem Turbulent Kinetic Energy ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = Lumped_User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

Lumped_User_Function

Syntax

Source For Cvfem Turbulent Kinetic Energy ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = Lumped_User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1
toggle	"string"	–

Nd_Table

Syntax

Source For Cvfem Turbulent Kinetic Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Source For Cvfem Turbulent Kinetic Energy ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The source, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Source For Cvfem Turbulent Kinetic Energy ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = Scalar_String_Function F = *f*

Summary

Value from a string function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
f	"string"	–

Sst_Turb_Ke

Syntax

Source For Cvfem Turbulent Kinetic Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Sst_Turb_Ke ()

Summary

K-omega (SST) turbulent source for CVFEM turbulent kinetic energy equation

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–

User_Field

Syntax

Source For Cvfem Turbulent Kinetic Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name	“string”	–
scaling	real	1
global_var	“string”	–
time_function	“string”	–

User_Function

Syntax

Source For Cvfem Turbulent Kinetic Energy ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Source For Cvfem Turbulent Kinetic Energy ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = User_Plugin (Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
Name	"string"	–
plugin_parameters	"string"	–

Vrtm

Syntax

Source For Cvfem Turbulent Kinetic Energy ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = Vrtm ()

Summary

Moving mesh source for conservative form, tke

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–

Cvfem Velocity Pressure Projection (

Constant

Syntax

Source For Cvfem Velocity Pressure Projection ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = Constant Value = *value*

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
value	real	–

Cvfem_Grad_Delta_P

Syntax

Source For Cvfem Velocity Pressure Projection ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = Cvfem_Grad_Delta_P ()

Summary

Grad Delta P source for CVFEM velocity pressure projection equation

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–

Exponential

Syntax

Source For Cvfem Velocity Pressure Projection ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The source, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Source For Cvfem Velocity Pressure Projection ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = Global (Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
global_name	"string"	–
component	integer	0

Lumped_Constant

Syntax

Source For Cvfem Velocity Pressure Projection ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = Lumped_Constant Value = *value*

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
value	real	–

Lumped_Exponential

Syntax

Source For Cvfem Velocity Pressure Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Lumped_Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
variable	“string”	–
constant	real	0
multiplier	real	1
exponent	real	–

Lumped_Global

Syntax

Source For Cvfem Velocity Pressure Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Lumped_Global (Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
global_name	“string”	–
component	integer	0

Lumped_Nd_Table

Syntax

Source For Cvfem Velocity Pressure Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Lumped_Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Lumped_Polynomial

Syntax

Source For Cvfem Velocity Pressure Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Lumped_Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Lumped_Scalar_String_Function

Syntax

Source For Cvfem Velocity Pressure Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Lumped_Scalar_String_Function F = *f*

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be

a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
f	"string"	–

Lumped_User_Field

Syntax

Source For Cvfem Velocity Pressure Projection ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = Lumped_User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

Lumped_User_Function

Syntax

Source For Cvfem Velocity Pressure Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Lumped_User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

Nd_Table

Syntax

Source For Cvfem Velocity Pressure Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Source For Cvfem Velocity Pressure Projection ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The source, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Source For Cvfem Velocity Pressure Projection ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = Scalar_String_Function F = *f*

Summary

Value from a string function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
f	"string"	–

User_Field

Syntax

Source For Cvfem Velocity Pressure Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name	“string”	–
scaling	real	1
global_var	“string”	–
time_function	“string”	–

User_Function

Syntax

Source For Cvfem Velocity Pressure Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1
toggle	"string"	–

User_Plugin

Syntax

Source For Cvfem Velocity Pressure Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = User_Plugin (Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
Name	"string"	–
plugin_parameters	"string"	–

Darcy Momentum (

Body_Acceleration

Syntax

Source For Darcy Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Body_Acceleration ()

Summary

Body acceleration drag source for darcy momentum

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–

Constant_Vector

Syntax

Source For Darcy Momentum ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = Constant_Vector ([Valuex = *valuex*] | [Valuey = *valuey*] | [Valuez = *valuez*])

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
valuex	real	0
valuey	real	0
valuez	real	0

Forchheimer_Drag

Syntax

Source For Darcy Momentum ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = Forchheimer_Drag ()

Summary

Forchheimer drag source for darcy momentum

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–

User_Field

Syntax

Source For Darcy Momentum ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a Vector User Field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name	“string”	–
scaling	real	1
global_var	“string”	–
time_function	“string”	–

User_Function

Syntax

Source For Darcy Momentum ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = User_Function ([Name_X = *name_x*] | [Name_Y = *name_y*] | [Name_Z = *name_z*] | X = *x* | [Multiplier = *multiplier*])

Summary

Value from a Vector User Function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name_x	"string"	–
name_y	"string"	–
name_z	"string"	–
x	"string"	–
multiplier	real	1

Vector_String_Function

Syntax

Source For Darcy Momentum ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = Vector_String_Function (F_X = *f_x* | F_Y = *f_y* | [F_Z = *f_z*])

Summary

Value from a string function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
f_x	"string"	–
f_y	"string"	–
f_z	"string"	–

Energy (

Bar_Elem_Field

Syntax

Source For Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Bar_Elem_Field ([Power_Output = *power_output*] | [Toggle = *toggle*] | Name = *name* | [Multiplier = *multiplier*])

Summary

This model has been deprecated and removed from the code.

Description

warning{This capability has been deprecated and removed}

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output

If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
power_output	“string”	–
toggle	“string”	–
name	“string”	–
multiplier	real	1

Burn_Front

Syntax

Source For Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Burn_Front ([Power_Output = *power_output*] | [Toggle = *toggle*])

Summary

Level_Set_Burn source for energy equation

Description

$$S = -\rho E \frac{\partial H(\phi)}{\partial t}$$

where ρ is the density, E the "BURN_HEAT_RELEASE" model, and $H(\phi)$ is the level-set Heaviside function.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output

If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
power_output	"string"	–
toggle	"string"	–

Calore_User_Sub

Syntax

Source For Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Calore_User_Sub ([Power_Output = *power_output*] | [Toggle = *toggle*] | Name = *name* | Type = *type* | [Multiplier = *multiplier*] | [Material_Data_Block = *material_data_block*] | [Data = *data*] | [Scaling_Field = *scaling_field*])

Summary

Energy source from a Calore user subroutine

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output

If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
power_output	“string”	–
toggle	“string”	–
name	“string”	–
type	“string”	–
multiplier	real	1
material_data_block	“string”	–
data	“string”	–
scaling_field	“string”	–

Chemeq_Heating

Syntax

Source For Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Chemeq_Heating ([Power_Output = *power_output*] | [Toggle = *toggle*] | [Model = *model*])

Summary

CHEMEQ heating source for energy equation

Description

See *Chemistry* for details on how the chemistry heating term is computed. Listing a model is optional, it will be retrieved automatically if omitted.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output

If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
power_output	“string”	–
toggle	“string”	–
model	“string”	–

Chemistry

Syntax

Source For Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Chemistry ([Power_Output = *power_output*] | [Toggle = *toggle*] | Mechanism = *mechanism*)

Summary

Chemical reaction source for energy equation

Description

See *Chemistry* for details.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output

If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
power_output	“string”	–
toggle	“string”	–
mechanism	“string”	–

Compressive_Work

Syntax

Source For Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Compressive_Work ([Power_Output = *power_output*] | [Toggle = *toggle*] | [Multiplier = *multiplier*])

Summary

Compressive work source for energy equation

Description

$$S = -MP\nabla \cdot v$$

where M is the optional multiplier, P is the pressure, and v the velocity.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output

If specified, the total (integrated) power of the given BC/Src is

calculated and stored into the global variable name specified as the *power_output* argument.

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
power_output	"string"	–
toggle	"string"	–
multiplier	real	1

Constant

Syntax

Source For Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Constant ([Power_Output = *power_output*] | [Toggle = *toggle*] | Value = *value*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output

If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
power_output	"string"	–
toggle	"string"	–
value	real	–

Curing_Foam_Heat_Of_Rxn

Syntax

Source For Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Curing_Foam_Heat_Of_Rxn ([Power_Output = *power_output*] | [Toggle = *toggle*] | H_Rxn = *h_rxn* | Mass_Fraction_Epoxy = *Mass_Fraction_Epoxy* | [Extent_Subindex = *extent_subindex*] | [Extent_Species_Name = *extent_species_name*])

Summary

Curing foam heat of reaction source for energy equation

Description

$$S = H_{\text{rxn}} Y_{\text{Epoxy}} \rho S_{\text{species}}$$

where H_{rxn}, and Y_{Epoxy} are the provided parameters, ρ is the density, and S_{species} is the species source. This model is intended to be used in conjunction with the "CURING_FOAM_EXTENT" species source model (*Curing_Foam_Extent*).

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output

If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
power_output	“string”	–
toggle	“string”	–
h_rxn	real	–
Mass_Fraction_Epoxy	real	–
extent_subindex	integer	-1
extent_species_name	“string”	–

Curing_Foam_Latent_Heat

Syntax

Source For Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Curing_Foam_Latent_Heat ([Power_Output = *power_output*] | [Toggle = *toggle*] | H_Evap = *h_evap* | Massfraction_Epoxy = *MassFraction_Epoxy* | Massfraction_Air = *MassFraction_Air* | Densityss_Fluorvapor = *DensitySS_FluorVapor* | Densityss_Air = *DensitySS_Air* | Densityss_Fluorliquid = *DensitySS_FluorLiquid* | Densityss_Epoxy = *DensitySS_Epoxy*)

Summary

Curing foam latent heat source for energy equation

Description

$$S = -\frac{H_{\text{evap}} N}{D} \frac{\partial \rho}{\partial t}$$

where H_{evap} is the provided parameter, ρ the density, and:

$$N = -\left(\frac{Y_e}{\rho_e} + \frac{1 - Y_e - Y_a}{\rho_{FL} + \frac{Y_A}{\rho_A}} \right)$$

$$D = \frac{1}{\rho_{FV}} - \frac{1}{\rho_{FL}}$$

where Y_e is the “MassFraction_Epoxy” parameter, ρ_e the “DensitySS_Epoxy” parameter, Y_a the “MassFraction_Air” parameter, ρ_{FL} the “DensitySS_FluorLiquid” parameter, and ρ_{FV} the “DensitySS_FluorVapor” parameter.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output

If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
power_output	"string"	–
toggle	"string"	–
h_evap	real	–
MassFraction_Epoxy	real	–
MassFraction_Air	real	–
DensitySS_FluorVapor	real	–
DensitySS_Air	real	–
DensitySS_FluorLiquid	real	–
DensitySS_Epoxy	real	–

Electrode_Object

Syntax

Source For Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Electrode_Object ([Power_Output = *power_output*] | [Toggle = *toggle*])

Summary

Electrode object source for energy equation to be used in some battery models.

Description

Uses the Zuzax library to evaluate electrochemical reactions within each element and compute source terms.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output

If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
power_output	"string"	–
toggle	"string"	–

Electrode_Object_Old

Syntax

Source For Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Electrode_Object_Old ([Power_Output = *power_output*] | [Toggle = *toggle*] | F = *f* | Electrodefile = *electrodeFile* | Kmolconversion = *kmolConversion* | [Jconversion = *JConversion*] | [Meterconversion = *meterConversion*] | [Max_Sub_Timesteps = *max_sub_timesteps*] | [Base_Delta = *base_delta*])

Summary

Deprecated, prefer ELECTRODE_OBJECT.

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output

If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
power_output	"string"	–
toggle	"string"	–
f	real	–
electrodeFile	"string"	–
kmolConversion	real	–
JConversion	real	1
meterConversion	real	1
max_sub_timesteps	integer	100
base_delta	real	1e-05

Exponential

Syntax

Source For Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Exponential ([Power_Output = *power_output*] | [Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The source, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output

If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
power_output	"string"	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Fin_Convective

Syntax

Source For Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Fin_Convective ([Power_Output = *power_output*] | [Toggle = *toggle*] | T_Inf = *t_inf* | H = *h*)

Summary

Fin Convective heat source for energy equation

Description

$$S = -HP\frac{T - T_\infty}{A}$$

where H and T_∞ are the provided parameters, A is the "BAR_AREA" model and P is the "BAR_PERIMITER" model.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output

If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
power_output	"string"	–
toggle	"string"	–
t_inf	real	–
h	real	–

Fin_Radiative

Syntax

Source For Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Fin_Radiative ([Power_Output = *power_output*] | [Toggle = *toggle*] | T_Inf = *t_inf* | [F = *f*])

Summary

Fin Radiative heat source for energy equation

Description

$$S = -EF\sigma P\frac{T^4 - T_\infty^4}{A}$$

where F and T_∞ are the provided parameters, σ is the Stefan-Boltzmann constant, A is the "BAR_AREA" model and P is the "BAR_PERIMITER" model.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output

If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
power_output	"string"	–
toggle	"string"	–
t_inf	real	–
f	real	1

Foam_Expansion

Syntax

Source For Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Foam_Expansion ([Power_Output = *power_output*] | [Toggle = *toggle*] | [Multiplier = *multiplier*])

Summary

Foam expansion source for energy equation

Description

$$S = -M\rho C_p T X \nabla \cdot v$$

where M is the "multiplier" parameter, ρ the density, c_p the specific heat, T the temperature, v the velocity, and X the volume fraction of gas.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output

If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
power_output	"string"	–
toggle	"string"	–
multiplier	real	-1

Fortran

Syntax

Source For Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Fortran ([Power_Output = *power_output*] | [Toggle = *toggle*] | [Multiplier = *multiplier*] | Sub_Name = *sub_name* | [Real_Data = *real_data*] | [Int_Data = *int_data*] | [Resource_Name = *resource_name*] | [Data = *data*])

Summary

Fortran subroutine energy source

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output

If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
power_output	"string"	–
toggle	"string"	–
multiplier	real	1
sub_name	"string"	–
real_data	"string"	–
int_data	"string"	–
resource_name	"string"	–
data	"string"	–

Full_Viscous_Dissipation

Syntax

Source For Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Full_Viscous_Dissipation ([Power_Output = *power_output*] | [Toggle = *toggle*] | [Multiplier = *multiplier*])

Summary

Full viscous dissipation source for energy equation

Description

$$S = M\sigma_v \cdot \nabla v$$

where M is the optional multiplier parameter, σ_v the stress tensor excluding the pressure contribution, and v the velocity.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output

If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
power_output	"string"	–
toggle	"string"	–
multiplier	real	1

General_Chem_Mass

Syntax

Source For Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = General_Chem_Mass ([Power_Output = *power_output*] | [Toggle = *toggle*] | Mechanism = *mechanism*)

Summary

Source term for mass-based general chemistry specification

Description

See *Chemistry* for details.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output

If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
power_output	"string"	–
toggle	"string"	–
mechanism	"string"	–

Global

Syntax

Source For Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Global ([Power_Output = *power_output*] | [Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output

If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
power_output	"string"	–
toggle	"string"	–
global_name	"string"	–
component	integer	0

Joule_Heating

Syntax

Source For Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Joule_Heating ([Power_Output = *power_output*] | [Toggle = *toggle*] | [Voltage_Subindex = *voltage_subindex*])

Summary

Joule heating source for energy equation

Description

$$S = -j \cdot \nabla V$$

where j is the current density and V the voltage.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output

If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
power_output	"string"	–
toggle	"string"	–
voltage_subindex	integer	-1

Joule_Heating_Material_Phase_Sum

Syntax

Source For Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Joule_Heating_Material_Phase_Sum ([Power_Output = *power_output*] | [Toggle = *toggle*] | [Voltage_Subindex = *voltage_subindex*])

Summary

Joule heating source for energy equation with current in multiple material phases.

Description

$$S = \sum_{\text{phases}} -j_p \cdot \nabla V_p$$

where j_p is the current density in the material phase and V_p the voltage in the material phase. This is most commonly used for battery models.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output

If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
power_output	"string"	–
toggle	"string"	–
voltage_subindex	integer	-1

Laser_Beer_Lambert

Syntax

Source For Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Laser_Beer_Lambert ([Power_Output = *power_output*] | [Toggle = *toggle*] | Beam_Diameter = *beam_diameter* | [Depth_Direction = *depth_direction*] | [Depth_Sign = *depth_sign*] | [Efficiency = *efficiency*] | R_Eff = *r_eff* | [Spatial_Influence_Factor = *spatial_influence_factor*] | [Old_Temperature = *old_temperature*] | Path_Function = *path_function* | [Deposition = *deposition*] | [Integrated = *integrated*] | [Clip = *clip*])

Summary

Laser Beer Lambert source for energy equation

Description

See *Directed Energy* for the appropriate syntax to use laser sources.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output

If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
power_output	“string”	–
toggle	“string”	–
beam_diameter	real	–
depth_direction	“string”	–
depth_sign	real	1
efficiency	real	1
r_eff	real	–
spatial_influence_factor	real	0
old_temperature	“string”	TRUE
path_function	“string”	–
deposition	“string”	FALSE
integrated	“string”	FALSE
clip	“string”	TRUE

Laser_Conical

Syntax

Source For Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Laser_Conical ([Power_Output = *power_output*] | [Toggle = *toggle*] | [Depth_Direction = *depth_direction*] | [Depth_Sign = *depth_sign*] | Penetration_Depth = *penetration_depth* | Beam_Diameter = *beam_diameter* | Beam_Diameter_At_Depth = *beam_diameter_at_depth* | [Efficiency = *efficiency*] | [Old_Temperature = *old_temperature*] | Path_Function = *path_function* | [Deposition = *deposition*] | [Integrated = *integrated*] | [Clip = *clip*])

Summary

Laser Conical source for energy equation

Description

See *Directed Energy* for the appropriate syntax to use laser sources.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output

If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
power_output	"string"	–
toggle	"string"	–
depth_direction	"string"	–
depth_sign	real	1
penetration_depth	real	–
beam_diameter	real	–
beam_diameter_at_depth	real	–
efficiency	real	1
old_temperature	"string"	TRUE
path_function	"string"	–
deposition	"string"	FALSE
integrated	"string"	FALSE
clip	"string"	TRUE

Laser_Double_Ellipsoidal

Syntax

Source For Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Laser_Double_Ellipsoidal ([Power_Output = *power_output*] | [Toggle = *toggle*] | Depth_Direction = *depth_direction* | [Depth_Sign = *depth_sign*] | Fore_Depth_Semi_Axis = *fore_depth_semi_axis* | Fore_Width_Semi_Axis = *fore_width_semi_axis* | Aft_Depth_Semi_Axis = *aft_depth_semi_axis* | Aft_Width_Semi_Axis = *aft_width_semi_axis* | Fore_Travel_Semi_Axis = *fore_travel_semi_axis* | Aft_Travel_Semi_Axis = *aft_travel_semi_axis* | Fore Power Fraction = *fore power fraction* | [Efficiency = *efficiency*] | [Old_Temperature = *old_temperature*] | Path_Function = *path_function* | [Deposition = *deposition*] | [Integrated = *integrated*] | [Clip = *clip*])

Summary

Laser Ellipsoidal source for energy equation

Description

See *Directed Energy* for the appropriate syntax to use laser sources.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output

If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
power_output	“string”	–
toggle	“string”	–
depth_direction	“string”	–
depth_sign	real	1
fore_depth_semi_axis	real	–
fore_width_semi_axis	real	–
aft_depth_semi_axis	real	–
aft_width_semi_axis	real	–
fore_travel_semi_axis	real	–
aft_travel_semi_axis	real	–
fore power fraction	real	–
efficiency	real	1
old_temperature	“string”	TRUE
path_function	“string”	–
deposition	“string”	FALSE
integrated	“string”	FALSE
clip	“string”	TRUE

Laser_Ellipsoidal

Syntax

Source For Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Laser_Ellipsoidal ([Power_Output = *power_output*] | [Toggle = *toggle*] | Depth_Direction = *depth_direction* | [Depth_Sign = *depth_sign*] | Travel_Semi_Axis = *travel_semi_axis* | Width_Semi_Axis = *width_semi_axis* | [Efficiency = *efficiency*] | [Old_Temperature = *old_temperature*] | Path_Function = *path_function* | [Deposition = *deposition*] | [Integrated = *integrated*] | [Clip = *clip*])

Summary

Laser Ellipsoidal source for energy equation

Description

See *Directed Energy* for the appropriate syntax to use laser sources.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output

If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
power_output	"string"	–
toggle	"string"	–
depth_direction	"string"	–
depth_sign	real	1
travel_semi_axis	real	–
width_semi_axis	real	–
efficiency	real	1
old_temperature	"string"	TRUE
path_function	"string"	–
deposition	"string"	FALSE
integrated	"string"	FALSE
clip	"string"	TRUE

Laser_Hemisphere

Syntax

Source For Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Laser_Hemisphere ([Power_Output = *power_output*] | [Toggle = *toggle*] | Beam_Diameter = *beam_diameter* | [Efficiency = *efficiency*] | [Distribution = *distribution*] | [Spatial_Influence_Factor = *spatial_influence_factor*] | [Depth_Direction = *depth_direction*] | [Depth_Sign = *depth_sign*] | [Old_Temperature = *old_temperature*] | Path_Function = *path_function* | [Deposition = *deposition*] | [Integrated = *integrated*] | [Clip = *clip*])

Summary

Laser Hemisphere source for energy equation

Description

See *Directed Energy* for the appropriate syntax to use laser sources.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output

If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the

power_output argument.

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
power_output	"string"	–
toggle	"string"	–
beam_diameter	real	–
efficiency	real	1
distribution	"string"	GAUSSIAN
spatial_influence_factor	real	1
depth_direction	"string"	–
depth_sign	real	1
old_temperature	"string"	TRUE
path_function	"string"	–
deposition	"string"	FALSE
integrated	"string"	FALSE
clip	"string"	TRUE

Laser_Parallelepiped

Syntax

Source For Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Laser_Parallelepiped ([Power_Output = *power_output*] | [Toggle = *toggle*] | [Depth_Direction = *depth_direction*] | [Depth_Sign = *depth_sign*] | Penetration_Depth = *penetration_depth* | Travel_Semi_Axis = *travel_semi_axis* | Width_Semi_Axis = *width_semi_axis* | [Efficiency = *efficiency*] | [Spatial_Influence_Factor = *spatial_influence_factor*] | [Old_Temperature = *old_temperature*] | Path_Function = *path_function* | [Deposition = *deposition*] | [Integrated = *integrated*] | [Clip = *clip*])

Summary

Laser Parallelepiped source for energy equation

Description

See *Directed Energy* for the appropriate syntax to use laser sources.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output
If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Toggle
If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
power_output	"string"	–
toggle	"string"	–
depth_direction	"string"	–
depth_sign	real	1
penetration_depth	real	–
travel_semi_axis	real	–
width_semi_axis	real	–
efficiency	real	1
spatial_influence_factor	real	1
old_temperature	"string"	TRUE
path_function	"string"	–
deposition	"string"	FALSE
integrated	"string"	FALSE
clip	"string"	TRUE

Laser_Sphere

Syntax

Source For Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Laser_Sphere ([Power_Output = *power_output*] | [Toggle = *toggle*] | Beam_Diameter = *beam_diameter* | [Depth_Direction = *depth_direction*] | [Depth_Sign = *depth_sign*] | [Efficiency = *efficiency*] | [Distribution = *distribution*] |

[Spatial_Influence_Factor = *spatial_influence_factor*] | [Old_Temperature = *old_temperature*] | Path_Function = *path_function* | [Deposition = *deposition*] | [Integrated = *integrated*] | [Clip = *clip*])

Summary

Laser Sphere source for energy equation

Description

See *Directed Energy* for the appropriate syntax to use laser sources.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output

If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
power_output	“string”	–
toggle	“string”	–
beam_diameter	real	–
depth_direction	“string”	–
depth_sign	real	1
efficiency	real	1
distribution	“string”	GAUSSIAN
spatial_influence_factor	real	1
old_temperature	“string”	TRUE
path_function	“string”	–
deposition	“string”	FALSE
integrated	“string”	FALSE
clip	“string”	TRUE

Ls_Burn

Syntax

Source For Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Ls_Burn ([Power_Output = *power_output*] | [Toggle = *toggle*] | [A = *a*])

Summary

Level_Set_Burn source for energy equation

Description

$$S = -A\frac{\partial H(\phi)}{\partial t}$$

where A is the provided parameter and $H(\phi)$ is the level-set Heaviside function.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output

If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
power_output	“string”	–
toggle	“string”	–
a	real	1

Lumped_Burn_Front

Syntax

Source For Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Lumped_Burn_Front ([Power_Output = *power_output*] | [Toggle = *toggle*])

Summary

Level_Set_Burn source for energy equation

Description

Same as *Burn_Front* but evaluated in a lumped fashion.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output

If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
power_output	“string”	–
toggle	“string”	–

Lumped_Chemistry

Syntax

Source For Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Lumped_Chemistry ([Power_Output = *power_output*] | [Toggle = *toggle*] | Mechanism = *mechanism*)

Summary

Lumped chemical reaction source for energy equation

Description

See *Chemistry* for details.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output

If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
power_output	“string”	–
toggle	“string”	–
mechanism	“string”	–

Lumped_Constant

Syntax

Source For Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Lumped_Constant ([Power_Output = *power_output*] | [Toggle = *toggle*] | Value = *value*)

Summary

Constant value

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output

If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
power_output	"string"	–
toggle	"string"	–
value	real	–

Lumped_Electrode_Object

Syntax

Source For Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Lumped_Electrode_Object ([Power_Output = *power_output*] | [Toggle = *toggle*])

Summary

Electrode object source for energy equation to be used in some battery models.

Description

Same as *Electrode_Object* but evaluated in a lumped fashion.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output

If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
power_output	"string"	–
toggle	"string"	–

Lumped_Exponential

Syntax

Source For Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Lumped_Exponential ([Power_Output = *power_output*] | [Toggle = *toggle*] | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output

If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
power_output	"string"	–
toggle	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Lumped_General_Chem_Mass

Syntax

Source For Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Lumped_General_Chem_Mass ([Power_Output = *power_output*] | [Toggle = *toggle*] | Mechanism = *mechanism*)

Summary

Lumped source term for mass-based general chemistry specification

Description

See *Chemistry* for details.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output

If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
power_output	"string"	–
toggle	"string"	–
mechanism	"string"	–

Lumped_Global

Syntax

Source For Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Lumped_Global ([Power_Output = *power_output*] | [Toggle = *toggle*] | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output

If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
power_output	"string"	–
toggle	"string"	–
global_name	"string"	–
component	integer	0

Lumped_Ls_Burn

Syntax

Source For Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Lumped_Ls_Burn ([Power_Output = *power_output*] | [Toggle = *toggle*] | [A = *a*])

Summary

Lumped Level_Set_Burn source for energy equation

Description

Same as *Ls_Burn* but evaluated in a lumped fashion.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output

If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
power_output	"string"	–
toggle	"string"	–
a	real	1

Lumped_Nd_Table

Syntax

Source For Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Lumped_Nd_Table ([Power_Output = *power_output*] | [Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output

If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
power_output	“string”	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Lumped_Polynomial

Syntax

Source For Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Lumped_Polynomial ([Power_Output = *power_output*] | [Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output

If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
power_output	"string"	–
toggle	"string"	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Lumped_Scalar_String_Function

Syntax

Source For Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Lumped_Scalar_String_Function ([Power_Output = *power_output*] | [Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output

If specified, the total (integrated) power of the given BC/Src is

calculated and stored into the global variable name specified as the *power_output* argument.

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
power_output	"string"	–
toggle	"string"	–
f	"string"	–

Lumped_User_Field

Syntax

Source For Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Lumped_User_Field ([Power_Output = *power_output*] | [Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output

If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
power_output	"string"	–
toggle	"string"	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

Lumped_User_Function

Syntax

Source For Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Lumped_User_Function ([Power_Output = *power_output*] | [Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output

If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
power_output	“string”	–
toggle	“string”	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1

Melting

Syntax

Source For Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Melting ([Power_Output = *power_output*] | [Toggle = *toggle*] | [Ts = *Ts*] | [Tl = *Tl*] | [Latent_Heat = *latent_heat*] | [Profile_Shape = *profile_shape*] | [Add_Advection = *add_advection*])

Summary

Energy source term to represent the latent heat of melting of a material.

Description

Uses a Gaussian (default) or Square distribution based on solidus (TS) and liquidus (TL) temperatures to compute a liquid fraction f_L. The Gaussian option is set so that only one percent of the energy gets applied outside of this range. The source term is computed as:

$$\rho L \frac{f_L^{n+1} - f_L{}^n}{\Delta t}$$

Uses either the LATENT_HEAT (L) expression or a provided “latent_heat = X” argument for each source. If both are present, the provided argument is used. This also assumes that the LATENT_HEAT is provided per mass (this expression multiplies it by the density). The solidus and liquidus temperatures can either be provided inline or in the material model. If both are provided, the inline values are used. The ADD_ADVECTION option adds an additional term that tracks the advection of the solidification front, which is currently implemented as:

$$\rho L \frac{\partial f_L}{\partial T} \boldsymbol{u} \cdot \nabla T$$

If this option is enabled, users should be aware that this term may be subject to temporal errors where large enough time steps can advance the solidification front over and incorrectly withhold energy over many cells.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output

If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
power_output	"string"	–
toggle	"string"	–
Ts	real	0
Tl	real	0
latent_heat	real	0
profile_shape	"string"	gaussian
add_advection	integer	–

Nd_Table

Syntax

Source For Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Nd_Table ([Power_Output = *power_output*] | [Toggle = *toggle*] | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output

If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
power_output	“string”	–
toggle	“string”	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Pid_Controlled

Syntax

Source For Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Pid_Controlled ([Power_Output = *power_output*] | [Toggle = *toggle*] | [P = *p*] | [I = *i*] | [D = *d*] | [T_I = *t_i*] | [T_D = *t_d*] | [Band = *Band*] | [Filter_Tau = *Filter_Tau*] | [Filter_Number = *Filter_Number*] | Setpoint_Function_Name = *Setpoint_Function_Name* | Control_Variable = *Control_Variable* | [Enable_Clamping = *enable_clamping*] | Max_Output = *Max_Output*)

Summary

PID controlled energy source term

Description

The controller setpoint vs. time is defined in a user function, and the controller feedback point uses a global variable (e.g. a data probe or reduction postprocessor).

The derivative term uses a filtered error value, using a low pass filter whose time constant is specified by the user. Like a heater, this is only capable of providing a positive influx. Even if the controller output is negative, the applied energy source will be truncated at 0. Likewise, the applied energy source cannot exceed the specified max source.

The nominal scaled PID controller output (between 0 and 1) is

$$u(t) = K_p e(t) + K_i \int_0^t e(s)ds + K_d \frac{\partial e_f}{\partial t}$$

where it is important to note that the K_p coefficient does not scale the others unless you have set it using the Band argument. In some PID implementations, the K_i and K_d coefficients are internally multiplied by K_p so care must be taken to provide the correct coefficients. This output $u(t)$ which is from 0 to 1 is scaled by the specified Max_Output value to produce the applied energy source. This means you **should not** include the output magnitude scaling in your P, I, or D parameters. If this controller is meant to recreate a controller that commands power directly, one must normalize by the max power.

When a derivative gain is provided, the derivative action is calculated using a filtered error to avoid derivative kick. The `Filter_Tau` argument can be used to specify the first-order derivative filter time constant τ, with a default value of 1 otherwise. Alternatively, one can provide the `Filter_Number` as argument $\tau = \frac{1}{N}$ where

$$K_d \frac{N}{1 + N\frac{1}{s}} = K_d \frac{1}{\tau + \frac{1}{s}} = \frac{K_d s}{\tau s + 1}$$

The P, I, and D coefficients can be specified in a number of ways:

- Just give a 'Band' argument. This calculates $P = 1/Band$ and uses the defaults of $I = 0.01P$ and $D = 0.0$
- Give a 'Band' argument and non-defaults for I and/or D. This still multiplies whatever you give for I and D by P.
- Give a 'P' argument instead of 'Band'. NOTE: in this mode the I and D values are not scaled by P.

If you provide both P and Band, P is used, the Band argument is ignored, and I and D are not scaled.

Alternatively, the integral and/or derivative gains can be provided using the integral time T_i (`T_i` argument) and derivative time T_d (`T_d` argument) so that $I = P/T_i$ and $D = P * T_d$ (regardless if P was set via band arg or not).

If `ENABLE_CLAMPING` is set to true, a clamping anti-windup algorithm is applied to the integral action of the controller.

The default values for I and D are 0.01 and 0 if not provided. The setpoint, controller output, error, filtered error, and filtered error time derivative are all output to automatically created global variables for diagnostics and postprocessing.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output
: If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Toggle
: If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
power_output	“string”	–
toggle	“string”	–
p	real	0
i	real	0
d	real	0
t_i	real	0
t_d	real	0
Band	real	0
Filter_Tau	real	1
Filter_Number	real	1
Setpoint_Function_Name	“string”	–
Control_Variable	“string”	–
enable_clamping	integer	–
Max_Output	real	–

Pid_Controlled_Bidirectional

Syntax

Source For Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Pid_Controlled_Bidirectional ([Power_Output = *power_output*] | [Toggle = *toggle*] | [P = *p*] | [I = *i*] | [D = *d*] | [T_I = *t_i*] | [T_D = *t_d*] | [Band = *Band*] | [Filter_Tau = *Filter_Tau*] | [Filter_Number = *Filter_Number*] | Setpoint_Function_Name = *Setpoint_Function_Name* | Control_Variable = *Control_Variable* | [Enable_Clamping = *enable_clamping*] | Max_Cooling_Power = *Max_Cooling_Power* | Max_Heating_Power = *Max_Heating_Power*)

Summary

This is similar to PID_Controlled, however, the source is applied to cool and heat the volume instead of just heating it. Max_Cooling_Power and Max_Heating_Power should be positive values. See description for PID_CONTROLLED at *Pid_Controlled* for more details

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output

If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
power_output	“string”	–
toggle	“string”	–
p	real	0
i	real	0
d	real	0
t_i	real	0
t_d	real	0
Band	real	0
Filter_Tau	real	1
Filter_Number	real	1
Setpoint_Function_Name	“string”	–
Control_Variable	“string”	–
enable_clamping	integer	–
Max_Cooling_Power	real	–
Max_Heating_Power	real	–

Pid_Controlled_Cooler

Syntax

Source For Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Pid_Controlled_Cooler ([Power_Output = *power_output*] | [Toggle = *toggle*] | [P = *p*] | [I = *i*] | [D = *d*] | [T_I = *t_i*] | [T_D = *t_d*] | [Band = *Band*] | [Filter_Tau = *Filter_Tau*] | [Filter_Number = *Filter_Number*] | Setpoint_Function_Name = *Setpoint_Function_Name* | Control_Variable = *Control_Variable* | [Enable_Clamping = *enable_clamping*] | Max_Cooling_Power = *Max_Cooling_Power*)

Summary

This is similar to PID_Controlled, however, the source is applied to cool the volume instead of heat it. Max_Cooling_Power should be a positive value. See description for PID_CONTROLLED at *Pid_Controlled* for more details

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output

If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
power_output	“string”	–
toggle	“string”	–
p	real	0
i	real	0
d	real	0
t_i	real	0
t_d	real	0
Band	real	0
Filter_Tau	real	1
Filter_Number	real	1
Setpoint_Function_Name	“string”	–
Control_Variable	“string”	–
enable_clamping	integer	–
Max_Cooling_Power	real	–

Polynomial

Syntax

Source For Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Polynomial ([Power_Output = *power_output*] | [Toggle = *toggle*] | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The source, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output
: If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Toggle
: If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
power_output	"string"	–
toggle	"string"	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Porous_Decomposition

Syntax

Source For Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Porous_Decomposition ([Power_Output = *power_output*] | [Toggle = *toggle*] | E = *e* | A = *a* | Rho = *rho* | Delta_H = *delta_h* | Coef = *coef* | Pexp = *pexp* | R = *r* | Pref = *Pref* | Tref = *Tref* | Type = *type*)

Summary

Porous decomposition source for energy equation

Description

$$S = \Delta H \rho Y_{\text{eff}} A \exp \frac{-E}{R(T + T_{\text{ref}})} \left(\frac{P}{P_{\text{ref}}} \right)^{P}_{\text{exp}}$$

where ΔH, ρ, A, E, R, T_{ref}, P_{ref}, and P_{exp} are the specified model parameters, T is the temperature, P is the pressure, and Y_{eff} has multiple forms depending on

the “type” parameter provided:

$$Y_{\text{eff}} = \begin{cases} (1-C+0.01)\sqrt{-log1-C+0.01}, & \text{type} = 1 \\ (1-C+0.01)\left(-log1-C+0.01\right)^{\frac{2}{3}}, & \text{type} = 2 \\ (1-C+0.01)\left(-log1-C+0.01\right)^{\frac{3}{4}}, & \text{type} = 3 \\ (1-C+0.01)\left(-log1-C+0.01\right)^{\frac{2}{3}}, & \text{type} = 4 \\ (1-C+0.01)^{\frac{2}{3}}, & \text{type} = 5 \\ \sqrt{1-C+0.01}, & \text{type} = 6 \\ 1, & \text{type} = 7 \\ 1-C+0.01, & \text{type} = 8 \\ (1-C+0.01)^2, & \text{type} = 9 \\ (1-C+0.01)^3, & \text{type} = 10 \\ \frac{1}{C}, & \text{type} = 11 \\ \frac{1}{-\log 1-C+0.02}, & \text{type} = 12 \\ \frac{(1-C+0.02)^{\frac{2}{3}}}{1-(1-C+0.02)^{\frac{1}{3}}}, & \text{type} = 13 \\ \frac{1}{(1-C+0.02)^{\frac{-1}{3}}-1}, & \text{type} = 14 \\ C^{\frac{3}{4}}, & \text{type} = 15 \\ C^{\frac{2}{3}}, & \text{type} = 16 \\ C^{\frac{1}{2}}, & \text{type} = 17 \\ C^{\frac{-1}{2}}, & \text{type} = 18 \\ C, & \text{type} = 19 \\ C(1-C+0.02), & \text{type} = 20 \\ C^2(1-C+0.02), & \text{type} = 21 \\ C^{\frac{3}{2}}, & \text{type} = 22 \\ (1-C+0.01)0.01(C-0.01), & \text{type} = 23 \end{cases}$$

where C is the species value bounded to $[0, 1]$.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output
: If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Toggle
: If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
power_output	"string"	–
toggle	"string"	–
e	real	–
a	real	–
rho	real	–
delta_h	real	–
coef	real	–
pexp	real	–
r	real	–
Pref	real	–
Tref	real	–
type	integer	–

Pressure_Material_Derivative

Syntax

Source For Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Pressure_Material_Derivative ([Power_Output = *power_output*] | [Toggle = *toggle*] | [Multiplier = *multiplier*])

Summary

Pressure material derivative source for energy equation

Description

$$S = M\left(\frac{\partial P}{\partial t} + v \cdot \nabla P\right)$$

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output

If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
power_output	"string"	–
toggle	"string"	–
multiplier	real	1

Rte_Field

Syntax

Source For Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Rte_Field ([Power_Output = *power_output*] | [Toggle = *toggle*])

Summary

Energy source from a scalar flux field for coupling with a radiative transport equation.

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output

If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
power_output	"string"	–
toggle	"string"	–

Rte_Sp

Syntax

Source For Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Rte_Sp ([Power_Output = *power_output*] | [Toggle = *toggle*] | Order = *order*)

Summary

RTE source for energy equation with Simplified Spherical Harmonic RTE model

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output

If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
power_output	"string"	–
toggle	"string"	–
order	integer	–

Scalar_String_Function

Syntax

Source For Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Scalar_String_Function ([Power_Output = *power_output*] | [Toggle = *toggle*] | F = *f*)

Summary

Value from a string function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output

If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
power_output	“string”	–
toggle	“string”	–
f	“string”	–

User_Field

Syntax

Source For Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = User_Field ([Power_Output = *power_output*] | [Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output

If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
power_output	“string”	–
toggle	“string”	–
name	“string”	–
scaling	real	1
global_var	“string”	–
time_function	“string”	–

User_Field_Volume_Heating

Syntax

Source For Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = User_Field_Volume_Heating ([Power_Output = *power_output*] | [Toggle = *toggle*] | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

User field volume heating source for energy equation

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output

If specified, the total (integrated) power of the given BC/Src is

calculated and stored into the global variable name specified as the *power_output* argument.

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
power_output	"string"	–
toggle	"string"	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Source For Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = User_Function ([Power_Output = *power_output*] | [Toggle = *toggle*] | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output

If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
power_output	"string"	–
toggle	"string"	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1

User_Plugin

Syntax

Source For Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = User_Plugin ([Power_Output = *power_output*] | [Toggle = *toggle*] | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output

If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
power_output	“string”	–
toggle	“string”	–
Name	“string”	–
plugin_parameters	“string”	–

Viscous_Dissipation

Syntax

Source For Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Viscous_Dissipation ([Power_Output = *power_output*] | [Toggle = *toggle*] | [Multiplier = *multiplier*])

Summary

Viscous dissipation source for energy equation

Description

$$S = M\sigma_v \cdot \nabla v$$

where M is the optional multiplier parameter, σ_v the viscous stress tensor and v the velocity.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Power_Output

If specified, the total (integrated) power of the given BC/Src is calculated and stored into the global variable name specified as the *power_output* argument.

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
power_output	"string"	–
toggle	"string"	–
multiplier	real	1

Enthalpy (

Constant

Syntax

Source For Enthalpy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Constant Value = *value*

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
value	real	–

Exponential

Syntax

Source For Enthalpy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The source, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Source For Enthalpy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Global (Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
global_name	"string"	–
component	integer	0

Lumped_Constant

Syntax

Source For Enthalpy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Lumped_Constant Value = *value*

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
value	real	–

Lumped_Exponential

Syntax

Source For Enthalpy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Lumped_Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
variable	“string”	–
constant	real	0
multiplier	real	1
exponent	real	–

Lumped_Global

Syntax

Source For Enthalpy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Lumped_Global (Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
global_name	“string”	–
component	integer	0

Lumped_Nd_Table

Syntax

Source For Enthalpy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Lumped_Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Lumped_Polynomial

Syntax

Source For Enthalpy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Lumped_Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Lumped_Scalar_String_Function

Syntax

Source For Enthalpy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Lumped_Scalar_String_Function F = *f*

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
f	"string"	–

Lumped_User_Field

Syntax

Source For Enthalpy ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = Lumped_User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

Lumped_User_Function

Syntax

Source For Enthalpy ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = Lumped_User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

Nd_Table

Syntax

Source For Enthalpy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Source For Enthalpy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The source, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Porous_Decomposition

Syntax

Source For Enthalpy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Porous_Decomposition (E = *e* | A = *a* | Rho = *rho* | Delta_H = *delta_h* | Coef = *coef* | Pexp = *pexp* | R = *r* | Pref = *Pref* | Tref = *Tref* | Type = *type*)

Summary

Porous decomposition source for enthalpy equation

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
e	real	–
a	real	–
rho	real	–
delta_h	real	–
coef	real	–
pexp	real	–
r	real	–
Pref	real	–
Tref	real	–
type	integer	–

Scalar_String_Function

Syntax

Source For Enthalpy ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = Scalar_String_Function F = *f*

Summary

Value from a string function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
f	“string”	–

User_Field

Syntax

Source For Enthalpy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name	“string”	–
scaling	real	1
global_var	“string”	–
time_function	“string”	–

User_Function

Syntax

Source For Enthalpy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1
toggle	"string"	–

User_Plugin

Syntax

Source For Enthalpy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = User_Plugin (Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
Name	"string"	–
plugin_parameters	"string"	–

Extension Speed (

Constant

Syntax

Source For Extension Speed ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Constant Value = *value*

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
value	real	–

Exponential

Syntax

Source For Extension Speed ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The source, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
variable	“string”	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Source For Extension Speed ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Global (Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
global_name	“string”	–
component	integer	0

Lumped_Constant

Syntax

Source For Extension Speed ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Lumped_Constant Value = *value*

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
value	real	–

Lumped_Exponential

Syntax

Source For Extension Speed ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = Lumped_Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Lumped_Global

Syntax

Source For Extension Speed ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = Lumped_Global (Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
global_name	"string"	–
component	integer	0

Lumped_Nd_Table

Syntax

Source For Extension Speed ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Lumped_Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Lumped_Polynomial

Syntax

Source For Extension Speed ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Lumped_Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Lumped_Scalar_String_Function

Syntax

Source For Extension Speed ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = Lumped_Scalar_String_Function F = *f*

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
f	"string"	–

Lumped_User_Field

Syntax

Source For Extension Speed ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = Lumped_User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

Lumped_User_Function

Syntax

Source For Extension Speed ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = Lumped_User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

Nd_Table

Syntax

Source For Extension Speed ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Source For Extension Speed ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The source, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Source For Extension Speed ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = Scalar_String_Function F = *f*

Summary

Value from a string function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
f	"string"	–

User_Field

Syntax

Source For Extension Speed ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name	“string”	–
scaling	real	1
global_var	“string”	–
time_function	“string”	–

User_Function

Syntax

Source For Extension Speed ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1
toggle	"string"	–

User_Plugin

Syntax

Source For Extension Speed ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = User_Plugin (Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
Name	"string"	–
plugin_parameters	"string"	–

Level Set (

Constant

Syntax

Source For Level Set ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Constant Value = *value*

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
value	real	–

Exponential

Syntax

Source For Level Set ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The source, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
variable	“string”	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Source For Level Set ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Global (Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
global_name	“string”	–
component	integer	0

Lumped_Constant

Syntax

Source For Level Set ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Lumped_Constant Value = *value*

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
value	real	–

Lumped_Exponential

Syntax

Source For Level Set ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Lumped_Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Lumped_Global

Syntax

Source For Level Set ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Lumped_Global (Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
global_name	"string"	–
component	integer	0

Lumped_Nd_Table

Syntax

Source For Level Set ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Lumped_Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Lumped_Polynomial

Syntax

Source For Level Set ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Lumped_Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Lumped_Scalar_String_Function

Syntax

Source For Level Set ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Lumped_Scalar_String_Function F = *f*

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
f	"string"	–

Lumped_User_Field

Syntax

Source For Level Set ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Lumped_User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

Lumped_User_Function

Syntax

Source For Level Set ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Lumped_User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1
toggle	"string"	–

Nd_Table

Syntax

Source For Level Set ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Source For Level Set ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The source, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Source For Level Set ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Scalar_String_Function F = *f*

Summary

Value from a string function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
f	"string"	–

User_Field

Syntax

Source For Level Set ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name	“string”	–
scaling	real	1
global_var	“string”	–
time_function	“string”	–

User_Function

Syntax

Source For Level Set ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1
toggle	"string"	–

User_Plugin

Syntax

Source For Level Set ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = User_Plugin (Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
Name	"string"	–
plugin_parameters	"string"	–

Lubrication (

Constant

Syntax

Source For Lubrication ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Constant Value = *value*

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
value	real	–

Exponential

Syntax

Source For Lubrication ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The source, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
variable	“string”	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Source For Lubrication ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Global (Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
global_name	“string”	–
component	integer	0

Lumped_Constant

Syntax

Source For Lubrication ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Lumped_Constant Value = *value*

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
value	real	–

Lumped_Exponential

Syntax

Source For Lubrication ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Lumped_Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Lumped_Global

Syntax

Source For Lubrication ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Lumped_Global (Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
global_name	"string"	–
component	integer	0

Lumped_Nd_Table

Syntax

Source For Lubrication ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Lumped_Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Lumped_Polynomial

Syntax

Source For Lubrication ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Lumped_Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Lumped_Scalar_String_Function

Syntax

Source For Lubrication ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Lumped_Scalar_String_Function F = *f*

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
f	"string"	–

Lumped_User_Field

Syntax

Source For Lubrication ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Lumped_User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

Lumped_User_Function

Syntax

Source For Lubrication ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Lumped_User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1
toggle	"string"	–

Nd_Table

Syntax

Source For Lubrication ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Source For Lubrication ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The source, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Source For Lubrication ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = Scalar_String_Function F = *f*

Summary

Value from a string function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
f	"string"	–

User_Field

Syntax

Source For Lubrication ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name	“string”	–
scaling	real	1
global_var	“string”	–
time_function	“string”	–

User_Function

Syntax

Source For Lubrication ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1
toggle	"string"	–

User_Plugin

Syntax

Source For Lubrication ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = User_Plugin (Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
Name	"string"	–
plugin_parameters	"string"	–

Magnetic Flux (

Constant_Vector

Syntax

Source For Magnetic Flux ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Constant_Vector ([X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
x	real	0
y	real	0
z	real	0

User_Field

Syntax

Source For Magnetic Flux ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a Vector User Field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name	“string”	–
scaling	real	1
global_var	“string”	–
time_function	“string”	–

User_Function

Syntax

Source For Magnetic Flux ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = User_Function ([Name_X = *name_x*] | [Name_Y = *name_y*] | [Name_Z = *name_z*] | X = *x* | [Multiplier = *multiplier*])

Summary

Value from a Vector User Function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name_x	“string”	–
name_y	“string”	–
name_z	“string”	–
x	“string”	–
multiplier	real	1

Vector_String_Function

Syntax

Source For Magnetic Flux ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Vector_String_Function (F_X = *f_x* | F_Y = *f_y* | [F_Z = *f_z*])

Summary

Value from a string function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
f_x	"string"	–
f_y	"string"	–
f_z	"string"	–

Mass Balance (

Chemistry

Syntax

Source For Mass Balance ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = Chemistry Mechanism = *mechanism*

Summary

Source term for mass-based general chemistry specification

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
mechanism	"string"	–

Component_Solubility

Syntax

Source For Mass Balance ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = Component_Solubility (C1 = *c1* | C2 = *c2*)

Summary

This model has been deprecated and removed from the code

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
c1	real	–
c2	real	–

Constant

Syntax

Source For Mass Balance ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = Constant Value = *value*

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
value	real	–

Dissolution_To_Liquid

Syntax

Source For Mass Balance ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = Dissolution_To_Liquid (Species = *species* | Kappa = *kappa*)

Summary

Dissolution rate model for transport between liquid porous species and gas pressure mass balance equations.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
species	"string"	–
kappa	real	–

Electrode_Object_Old

Syntax

Source For Mass Balance ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = Electrode_Object_Old (F = *f* | Electrodefile = *electrodeFile* | Kmolconversion = *kmolConversion* | [Jconversion = *JConversion*] | [Meterconversion = *meterConversion*] | [Max_Sub_Timesteps = *max_sub_timesteps*] | [Base_Delta = *base_delta*])

Summary

Electrode object source for current equation to be used with porous electrodes

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
f	real	–
electrodeFile	"string"	–
kmolConversion	real	–
JConversion	real	1
meterConversion	real	1
max_sub_timesteps	integer	100
base_delta	real	1e-05

Exponential

Syntax

Source For Mass Balance ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The source, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

From_Current_Source

Syntax

Source For Mass Balance ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = From_Current_Source (Mult = *mult* | [F = *f*])

Summary

Generating species source from equivalent current source

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
mult	real	–
f	real	9.649e+04

Global

Syntax

Source For Mass Balance ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Global (Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
global_name	“string”	–
component	integer	0

Ionic_Electroneutrality

Syntax

Source For Mass Balance ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Ionic_Electroneutrality ()

Summary

Source ensuring electroneutrality of ions

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–

Lumped_Chemistry

Syntax

Source For Mass Balance ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Lumped_Chemistry Mechanism = *mechanism*

Summary

Lumped source term for mass-based general chemistry specification

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
mechanism	“string”	–

Lumped_Constant

Syntax

Source For Mass Balance ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Lumped_Constant Value = *value*

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
value	real	–

Lumped_Exponential

Syntax

Source For Mass Balance ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = Lumped_Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
variable	“string”	–
constant	real	0
multiplier	real	1
exponent	real	–

Lumped_General_Chem_Mass

Syntax

Source For Mass Balance ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = Lumped_General_Chem_Mass Mechanism = *mechanism*

Summary

Lumped source term for mass-based general chemistry specification

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
mechanism	"string"	–

Lumped_Global

Syntax

Source For Mass Balance ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = Lumped_Global (Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
global_name	"string"	–
component	integer	0

Lumped_Nd_Table

Syntax

Source For Mass Balance ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = Lumped_Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Lumped_Polynomial

Syntax

Source For Mass Balance ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Lumped_Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Lumped_Scalar_String_Function

Syntax

Source For Mass Balance ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = Lumped_Scalar_String_Function F = *f*

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
f	"string"	–

Lumped_User_Field

Syntax

Source For Mass Balance ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = Lumped_User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

Lumped_User_Function

Syntax

Source For Mass Balance ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = Lumped_User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1
toggle	"string"	–

Nd_Table

Syntax

Source For Mass Balance ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Source For Mass Balance ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The source, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Porous_Decomposition

Syntax

Source For Mass Balance ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = Porous_Decomposition (E = *e* | A = *a* | Rho = *rho* | Delta_H = *delta_h* | Coef = *coef* | Pexp = *pexp* | R = *r* | Pref = *Pref* | Tref = *Tref* | Type = *type*)

Summary

Porous decomposition source for momentum equation

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
e	real	–
a	real	–
rho	real	–
delta_h	real	–
coef	real	–
pexp	real	–
r	real	–
Pref	real	–
Tref	real	–
type	integer	–

Pressurization_Model

Syntax

Source For Mass Balance ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = Pressurization_Model Model_Name = *model_name*

Summary

Gas mass source for use on pressurization model bulk node.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
model_name	“string”	–

Pressurization_Model_Venting

Syntax

Source For Mass Balance ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Pressurization_Model_Venting ()

Summary

Automatically created source term to apply venting boundary condition to pressurization model bulk node.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–

Scalar_String_Function

Syntax

Source For Mass Balance ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Scalar_String_Function F = *f*

Summary

Value from a string function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
f	“string”	–

User_Field

Syntax

Source For Mass Balance ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name	“string”	–
scaling	real	1
global_var	“string”	–
time_function	“string”	–

User_Function

Syntax

Source For Mass Balance ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1
toggle	"string"	–

User_Plugin

Syntax

Source For Mass Balance ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = User_Plugin (Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
Name	"string"	–
plugin_parameters	"string"	–

Mesh (

Constant_Vector

Syntax

Source For Mesh ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Constant_Vector ([Valuex = *valuex*] | [Valuey = *valuey*] | [Valuez = *valuez*])

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
valuex	real	0
valuey	real	0
valuez	real	0

Hydrostatic

Syntax

Source For Mesh ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = Hydrostatic ([Gx = *gx*] | [Gy = *gy*] | [Gz = *gz*] | [Ref_Density = *ref_density*] | [Density_Phase = *density_phase*])

Summary

Hydrostatic source for MESH equation. Input must also include DENSITY_PHASE = SOLID

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
gx	real	0
gy	real	0
gz	real	0
ref_density	real	0
density_phase	“string”	–

Mesh_Body_Acceleration

Syntax

Source For Mesh ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Mesh_Body_Acceleration ()

Summary

Body acceleration source for MESH equation

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–

User_Field

Syntax

Source For Mesh ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a Vector User Field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name	“string”	–
scaling	real	1
global_var	“string”	–
time_function	“string”	–

User_Function

Syntax

Source For Mesh ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = User_Function ([Name_X = *name_x*] | [Name_Y = *name_y*] | [Name_Z = *name_z*] | X = *x* | [Multiplier = *multiplier*])

Summary

Value from a Vector User Function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name_x	“string”	–
name_y	“string”	–
name_z	“string”	–
x	“string”	–
multiplier	real	1

Vector_String_Function

Syntax

Source For Mesh ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Vector_String_Function (F_X = *f_x* | F_Y = *f_y* | [F_Z = *f_z*])

Summary

Value from a string function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
f_x	"string"	–
f_y	"string"	–
f_z	"string"	–

Mesh Body Acceleration (

Constant_Vector

Syntax

Source For Mesh Body Acceleration ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = Constant_Vector ([Valuex = *valuex*] | [Valuey = *valuey*] | [Valuez = *valuez*])

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
valuex	real	0
valuey	real	0
valuez	real	0

User_Field

Syntax

Source For Mesh Body Acceleration ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a Vector User Field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name	“string”	–
scaling	real	1
global_var	“string”	–
time_function	“string”	–

User_Function

Syntax

Source For Mesh Body Acceleration ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = User_Function ([Name_X = *name_x*] | [Name_Y = *name_y*] | [Name_Z = *name_z*] | X = *x* | [Multiplier = *multiplier*])

Summary

Value from a Vector User Function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name_x	"string"	–
name_y	"string"	–
name_z	"string"	–
x	"string"	–
multiplier	real	1

Vector_String_Function

Syntax

Source For Mesh Body Acceleration ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = Vector_String_Function (F_X = *f_x* | F_Y = *f_y* | [F_Z = *f_z*])

Summary

Value from a string function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
f_x	"string"	–
f_y	"string"	–
f_z	"string"	–

Momentum (

Bf_Hydrostatic

Syntax

Source For Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Bf_Hydrostatic ([Gx = *gx*] | [Gy = *gy*] | [Gz = *gz*] | [Ref_Density = *ref_density*] | [Density_Phase = *density_phase*])

Summary

Balanced-force Hydrostatic source for momentum equation

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
gx	real	0
gy	real	0
gz	real	0
ref_density	real	0
density_phase	“string”	–

Body_Acceleration

Syntax

Source For Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Body_Acceleration ()

Summary

Body acceleration drag source for momentum equation

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–

Boussinesq

Syntax

Source For Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Boussinesq ([Gx = *gx*] | [Gy = *gy*] | [Gz = *gz*] | Temp_Ref = *temp_ref*)

Summary

Boussinesq source for momentum equation

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
gx	real	0
gy	real	0
gz	real	0
temp_ref	real	–

Constant_Vector

Syntax

Source For Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Constant_Vector ([X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
x	real	0
y	real	0
z	real	0

Hydrostatic

Syntax

Source For Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Hydrostatic ([Gx = *gx*] | [Gy = *gy*] | [Gz = *gz*] | [Ref_Density = *ref_density*] | [Density_Phase = *density_phase*])

Summary

Hydrostatic source for momentum equation

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
gx	real	0
gy	real	0
gz	real	0
ref_density	real	0
density_phase	“string”	–

Lorentz

Syntax

Source For Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Lorentz [Magnetic_Field = *magnetic_field*]

Summary

Lorentz source due to either an externally applied or calculated magnetic field J X B. J is the current from the current equation. B is either an imported vector field whose name is given by MAGNETIC_FIELD or the magnetic field calculated by Aria

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
magnetic_field	"string"	–

Ls_Capillary

Syntax

Source For Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Ls_Capillary ()

Summary

LS capillary source for momentum equation

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–

Ls_Capillary_Gap

Syntax

Source For Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Ls_Capillary_Gap (V_W = *v_w* | G = *g* | Theta = *theta*)

Summary

LS capillary gap source for momentum equation

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
v_w	real	–
g	real	–
theta	real	–

Ls_Heaviside_Capillary

Syntax

Source For Momentum ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = Ls_Heaviside_Capillary ()

Summary

LS heaviside capillary source for momentum equation

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–

Potential

Syntax

Source For Momentum ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = Potential ()

Summary

Potential source for momentum equation

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–

Pressure_Gradient

Syntax

Source For Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Pressure_Gradient [Pressure = *pressure*]

Summary

Pressure gradient source for momentum equation

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
pressure	“string”	–

User_Field

Syntax

Source For Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a Vector User Field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Source For Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = User_Function ([Name_X = *name_x*] | [Name_Y = *name_y*] | [Name_Z = *name_z*] | X = *x* | [Multiplier = *multiplier*])

Summary

Value from a Vector User Function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name_x	"string"	–
name_y	"string"	–
name_z	"string"	–
x	"string"	–
multiplier	real	1

Vector_String_Function

Syntax

Source For Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Vector_String_Function (F_X = *f_x* | F_Y = *f_y* | [F_Z = *f_z*])

Summary

Value from a string function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
f_x	"string"	–
f_y	"string"	–
f_z	"string"	–

Viscous_Gap

Syntax

Source For Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Viscous_Gap ()

Summary

Viscous gap source for momentum equation

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–

Nacl Mass Balance (

Constant

Syntax

Source For Nacl Mass Balance ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Constant Value = *value*

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
value	real	–

Exponential

Syntax

Source For Nacl Mass Balance ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The source, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Source For Nacl Mass Balance ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Global (Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
global_name	"string"	–
component	integer	0

Lumped_Constant

Syntax

Source For Nacl Mass Balance ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Lumped_Constant Value = *value*

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
value	real	–

Lumped_Exponential

Syntax

Source For Nacl Mass Balance ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Lumped_Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
variable	“string”	–
constant	real	0
multiplier	real	1
exponent	real	–

Lumped_Global

Syntax

Source For Nacl Mass Balance ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Lumped_Global (Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
global_name	“string”	–
component	integer	0

Lumped_Nd_Table

Syntax

Source For Nacl Mass Balance ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = Lumped_Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Lumped_Polynomial

Syntax

Source For Nacl Mass Balance ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Lumped_Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Lumped_Scalar_String_Function

Syntax

Source For Nacl Mass Balance ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Lumped_Scalar_String_Function F = *f*

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be

a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
f	"string"	–

Lumped_User_Field

Syntax

Source For Nacl Mass Balance ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Lumped_User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

Lumped_User_Function

Syntax

Source For Nacl Mass Balance ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Lumped_User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

Nd_Table

Syntax

Source For Nacl Mass Balance ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Source For Nacl Mass Balance ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The source, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Source For Nacl Mass Balance ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = Scalar_String_Function F = *f*

Summary

Value from a string function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
f	"string"	–

User_Field

Syntax

Source For Nacl Mass Balance ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name	“string”	–
scaling	real	1
global_var	“string”	–
time_function	“string”	–

User_Function

Syntax

Source For Nacl Mass Balance ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1
toggle	"string"	–

User_Plugin

Syntax

Source For Nacl Mass Balance ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = User_Plugin (Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
Name	"string"	–
plugin_parameters	"string"	–

Neutron (

Absorption

Syntax

Source For Neutron ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Absorption ()

Summary

Absorption source for neutron equation

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–

Constant

Syntax

Source For Neutron ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = Constant Value = *value*

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
value	real	–

Exponential

Syntax

Source For Neutron ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The source, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the

constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Fission

Syntax

Source For Neutron ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Fission (Beta = *beta* | N_F = *n_f*)

Summary

Fission source for neutron equation

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
beta	real	–
n_f	real	–

Global

Syntax

Source For Neutron ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Global (Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
global_name	“string”	–
component	integer	0

Lumped_Constant

Syntax

Source For Neutron ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Lumped_Constant Value = *value*

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
value	real	–

Lumped_Exponential

Syntax

Source For Neutron ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Lumped_Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Lumped_Global

Syntax

Source For Neutron ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Lumped_Global (Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
global_name	"string"	–
component	integer	0

Lumped_Nd_Table

Syntax

Source For Neutron ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Lumped_Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Lumped_Polynomial

Syntax

Source For Neutron ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Lumped_Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Lumped_Scalar_String_Function

Syntax

Source For Neutron ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Lumped_Scalar_String_Function F = *f*

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
f	"string"	–

Lumped_User_Field

Syntax

Source For Neutron ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Lumped_User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

Lumped_User_Function

Syntax

Source For Neutron ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Lumped_User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1
toggle	"string"	–

Nd_Table

Syntax

Source For Neutron ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Source For Neutron ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The source, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Precursor_Reaction

Syntax

Source For Neutron ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Precursor_Reaction Lambda = *lambda*

Summary

Precursor reaction source for neutron equation

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
lambda	real	–

Scalar_String_Function

Syntax

Source For Neutron ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Scalar_String_Function F = *f*

Summary

Value from a string function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
f	"string"	–

User_Field

Syntax

Source For Neutron ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Source For Neutron ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Source For Neutron ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = User_Plugin (Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
Name	"string"	–
plugin_parameters	"string"	–

Nonwetting Phase Mass Balance (

Constant

Syntax

Source For Nonwetting Phase Mass Balance ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = Constant Value = *value*

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
value	real	–

Exponential

Syntax

Source For Nonwetting Phase Mass Balance ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The source, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Source For Nonwetting Phase Mass Balance ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = Global (Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
global_name	"string"	–
component	integer	0

Lumped_Constant

Syntax

Source For Nonwetting Phase Mass Balance ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Lumped_Constant Value = *value*

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
value	real	–

Lumped_Exponential

Syntax

Source For Nonwetting Phase Mass Balance ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Lumped_Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
variable	“string”	–
constant	real	0
multiplier	real	1
exponent	real	–

Lumped_Global

Syntax

Source For Nonwetting Phase Mass Balance ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Lumped_Global (Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
global_name	“string”	–
component	integer	0

Lumped_Nd_Table

Syntax

Source For Nonwetting Phase Mass Balance ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Lumped_Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Lumped_Polynomial

Syntax

Source For Nonwetting Phase Mass Balance ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = Lumped_Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Lumped_Scalar_String_Function

Syntax

Source For Nonwetting Phase Mass Balance ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = Lumped_Scalar_String_Function F = *f*

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
f	"string"	–

Lumped_User_Field

Syntax

Source For Nonwetting Phase Mass Balance ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = Lumped_User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

Lumped_User_Function

Syntax

Source For Nonwetting Phase Mass Balance ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = Lumped_User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1
toggle	"string"	–

Nd_Table

Syntax

Source For Nonwetting Phase Mass Balance ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Source For Nonwetting Phase Mass Balance ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The source, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Porous_Decomposition

Syntax

Source For Nonwetting Phase Mass Balance ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = Porous_Decomposition (E = *e* | A = *a* | Rho = *rho* | Delta_H = *delta_h* | Coef = *coef* | Pexp = *pexp* | R = *r* | Pref = *Pref* | Tref = *Tref* | Type = *type*)

Summary

Porous decomposition source for momentum equation

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
e	real	–
a	real	–
rho	real	–
delta_h	real	–
coef	real	–
pexp	real	–
r	real	–
Pref	real	–
Tref	real	–
type	integer	–

Scalar_String_Function

Syntax

Source For Nonwetting Phase Mass Balance ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = Scalar_String_Function F = *f*

Summary

Value from a string function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
f	“string”	–

User_Field

Syntax

Source For Nonwetting Phase Mass Balance ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name	“string”	–
scaling	real	1
global_var	“string”	–
time_function	“string”	–

User_Function

Syntax

Source For Nonwetting Phase Mass Balance ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Source For Nonwetting Phase Mass Balance ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = User_Plugin (Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
Name	“string”	–
plugin_parameters	“string”	–

Porous Enthalpy (

Chemistry

Syntax

Source For Porous Enthalpy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Chemistry Mechanism = *mechanism*

Summary

Source term for mass-based general chemistry specification

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
mechanism	"string"	–

Compressible_Pressure

Syntax

Source For Porous Enthalpy ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = Compressible_Pressure ()

Summary

Porous compressible pressure contribution term

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–

Constant

Syntax

Source For Porous Enthalpy ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = Constant Value = *value*

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
value	real	–

Exponential

Syntax

Source For Porous Enthalpy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The source, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
variable	“string”	–
constant	real	0
multiplier	real	1
exponent	real	–

From_Mass_Balance_Source

Syntax

Source For Porous Enthalpy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = From_Mass_Balance_Source [Species = *species*]

Summary

Enthalpy source to maintain constant T when adding mass.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
species	"string"	–

General_Chem_Mass

Syntax

Source For Porous Enthalpy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = General_Chem_Mass Mechanism = *mechanism*

Summary

Source term for mass-based general chemistry specification

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
mechanism	"string"	–

Global

Syntax

Source For Porous Enthalpy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Global (Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
global_name	“string”	–
component	integer	0

Lumped_Chemistry

Syntax

Source For Porous Enthalpy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Lumped_Chemistry Mechanism = *mechanism*

Summary

Source term for mass-based general chemistry specification

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
mechanism	“string”	–

Lumped_Constant

Syntax

Source For Porous Enthalpy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Lumped_Constant Value = *value*

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
value	real	–

Lumped_Exponential

Syntax

Source For Porous Enthalpy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Lumped_Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
variable	“string”	–
constant	real	0
multiplier	real	1
exponent	real	–

Lumped_General_Chem_Mass

Syntax

Source For Porous Enthalpy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Lumped_General_Chem_Mass Mechanism = *mechanism*

Summary

Source term for mass-based general chemistry specification

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
mechanism	“string”	–

Lumped_Global

Syntax

Source For Porous Enthalpy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Lumped_Global (Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
global_name	“string”	–
component	integer	0

Lumped_Nd_Table

Syntax

Source For Porous Enthalpy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Lumped_Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Lumped_Polynomial

Syntax

Source For Porous Enthalpy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Lumped_Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Lumped_Scalar_String_Function

Syntax

Source For Porous Enthalpy ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = Lumped_Scalar_String_Function F = *f*

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
f	"string"	–

Lumped_User_Field

Syntax

Source For Porous Enthalpy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Lumped_User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

Lumped_User_Function

Syntax

Source For Porous Enthalpy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Lumped_User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1
toggle	"string"	–

Lumped_Volumetric_Transfer

Syntax

Source For Porous Enthalpy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Lumped_Volumetric_Transfer Remote_Phase = *remote_phase*

Summary

Porous volumetric heat transfer source

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
remote_phase	"string"	–

Nd_Table

Syntax

Source For Porous Enthalpy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Source For Porous Enthalpy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The source, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Source For Porous Enthalpy ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = Scalar_String_Function F = *f*

Summary

Value from a string function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
f	"string"	–

User_Field

Syntax

Source For Porous Enthalpy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name	“string”	–
scaling	real	1
global_var	“string”	–
time_function	“string”	–

User_Function

Syntax

Source For Porous Enthalpy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1
toggle	"string"	–

User_Plugin

Syntax

Source For Porous Enthalpy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = User_Plugin (Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
Name	"string"	–
plugin_parameters	"string"	–

Volumetric_Transfer

Syntax

Source For Porous Enthalpy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Volumetric_Transfer Remote_Phase = *remote_phase*

Summary

Porous volumetric heat transfer source

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
remote_phase	"string"	–

Porous Species (

Chemistry

Syntax

Source For Porous Species ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Chemistry Mechanism = *mechanism*

Summary

General chemistry porous species source.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
mechanism	"string"	–

Constant

Syntax

Source For Porous Species ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Constant Value = *value*

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
value	real	–

Dissolution_From_Gas

Syntax

Source For Porous Species ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Dissolution_From_Gas (Species = *species* | Kappa = *kappa*)

Summary

Dissolution rate model for transport between liquid porous species and gas pressure mass balance equations.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
species	“string”	–
kappa	real	–

Electrode_Object

Syntax

Source For Porous Species ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Electrode_Object ()

Summary

Electrode object source for species equation to be used with porous electrodes

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–

Electrode_Object_Old

Syntax

Source For Porous Species ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = Electrode_Object_Old (F = *f* | Electrodefile = *electrodeFile* | Kmolconversion = *kmolConversion* | [Jconversion = *JConversion*] | [Meterconversion = *meterConversion*] | [Max_Sub_Timesteps = *max_sub_timesteps*] | [Base_Delta = *base_delta*])

Summary

Electrode object source for species equation to be used with porous electrodes

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
f	real	–
electrodeFile	“string”	–
kmolConversion	real	–
JConversion	real	1
meterConversion	real	1
max_sub_timesteps	integer	100
base_delta	real	1e-05

Exponential

Syntax

Source For Porous Species ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The source, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

From_Current_Source

Syntax

Source For Porous Species ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = From_Current_Source (Mult = *mult* | [F = *f*])

Summary

Generating species source from equivalent current source

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
mult	real	–
f	real	9.649e+04

Global

Syntax

Source For Porous Species ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Global (Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
global_name	“string”	–
component	integer	0

Ionic_Electroneutrality

Syntax

Source For Porous Species ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Ionic_Electroneutrality ()

Summary

Enforcing electroneutrality for ion species

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–

Lumped_Chemistry

Syntax

Source For Porous Species ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = Lumped_Chemistry Mechanism = *mechanism*

Summary

General chemistry lumped porous species source.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
mechanism	“string”	–

Lumped_Constant

Syntax

Source For Porous Species ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = Lumped_Constant Value = *value*

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
value	real	–

Lumped_Electrode_Object

Syntax

Source For Porous Species ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Lumped_Electrode_Object ()

Summary

Electrode object source for species equation to be used with porous electrodes

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–

Lumped_Exponential

Syntax

Source For Porous Species ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Lumped_Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Lumped_Global

Syntax

Source For Porous Species ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Lumped_Global (Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
global_name	"string"	–
component	integer	0

Lumped_Nd_Table

Syntax

Source For Porous Species ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Lumped_Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Lumped_Polynomial

Syntax

Source For Porous Species ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Lumped_Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Lumped_Scalar_String_Function

Syntax

Source For Porous Species ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = Lumped_Scalar_String_Function F = *f*

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
f	"string"	–

Lumped_User_Field

Syntax

Source For Porous Species ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Lumped_User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

Lumped_User_Function

Syntax

Source For Porous Species ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Lumped_User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1
toggle	"string"	–

Nd_Table

Syntax

Source For Porous Species ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Source For Porous Species ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The source, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Source For Porous Species ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = Scalar_String_Function F = *f*

Summary

Value from a string function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
f	"string"	–

User_Field

Syntax

Source For Porous Species ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name	“string”	–
scaling	real	1
global_var	“string”	–
time_function	“string”	–

User_Function

Syntax

Source For Porous Species ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1
toggle	"string"	–

User_Plugin

Syntax

Source For Porous Species ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = User_Plugin (Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
Name	"string"	–
plugin_parameters	"string"	–

Potential (

Constant_Vector

Syntax

Source For Potential ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Constant_Vector ([X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
x	real	0
y	real	0
z	real	0

Hydrostatic

Syntax

Source For Potential ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = Hydrostatic ([Gx = *gx*] | [Gy = *gy*] | [Gz = *gz*] | [Ref_Density = *ref_density*] | [Density_Phase = *density_phase*])

Summary

Hydrostatic source for potential equation

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
gx	real	0
gy	real	0
gz	real	0
ref_density	real	0
density_phase	“string”	–

User_Field

Syntax

Source For Potential ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a Vector User Field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Source For Potential ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = User_Function ([Name_X = *name_x*] | [Name_Y = *name_y*] | [Name_Z = *name_z*] | X = *x* | [Multiplier = *multiplier*])

Summary

Value from a Vector User Function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name_x	"string"	–
name_y	"string"	–
name_z	"string"	–
x	"string"	–
multiplier	real	1

Vector_String_Function

Syntax

Source For Potential ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Vector_String_Function (F_X = *f_x* | F_Y = *f_y* | [F_Z = *f_z*])

Summary

Value from a string function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
f_x	"string"	–
f_y	"string"	–
f_z	"string"	–

Precursor Conservation (

Fission_Precursor_Concentration

Syntax

Source For Precursor Conservation ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Fission_Precursor_Concentration (Beta = *beta* | N_F = *n_f* | P_R = *p_r*)

Summary

Fission source for precursor conservation equation

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
beta	real	–
n_f	real	–
p_r	real	–

Precursor_Reaction_Concentration

Syntax

Source For Precursor Conservation ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Precursor_Reaction_Concentration Lambda = *lambda*

Summary

Precursor reaction concentration source for precursor conservation equation

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
lambda	real	–

Projected Nu Hat Density (

Constant

Syntax

Source For Projected Nu Hat Density ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Constant Value = *value*

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
value	real	–

Exponential

Syntax

Source For Projected Nu Hat Density ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The source, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Source For Projected Nu Hat Density ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = Global (Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
global_name	"string"	–
component	integer	0

Lumped_Constant

Syntax

Source For Projected Nu Hat Density ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = Lumped_Constant Value = *value*

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
value	real	–

Lumped_Exponential

Syntax

Source For Projected Nu Hat Density ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = Lumped_Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
variable	“string”	–
constant	real	0
multiplier	real	1
exponent	real	–

Lumped_Global

Syntax

Source For Projected Nu Hat Density ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = Lumped_Global (Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
global_name	“string”	–
component	integer	0

Lumped_Nd_Table

Syntax

Source For Projected Nu Hat Density ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = Lumped_Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Lumped_Polynomial

Syntax

Source For Projected Nu Hat Density ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = Lumped_Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
variable	“string”	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Lumped_Scalar_String_Function

Syntax

Source For Projected Nu Hat Density ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = Lumped_Scalar_String_Function F = *f*

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be

a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
f	"string"	–

Lumped_User_Field

Syntax

Source For Projected Nu Hat Density ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Lumped_User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

Lumped_User_Function

Syntax

Source For Projected Nu Hat Density ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Lumped_User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

Nd_Table

Syntax

Source For Projected Nu Hat Density ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Source For Projected Nu Hat Density ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The source, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Source For Projected Nu Hat Density ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = Scalar_String_Function F = *f*

Summary

Value from a string function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
f	"string"	–

User_Field

Syntax

Source For Projected Nu Hat Density ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Source For Projected Nu Hat Density ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Source For Projected Nu Hat Density ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = User_Plugin (Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
Name	“string”	–
plugin_parameters	“string”	–

Projected Sdr (

Constant

Syntax

Source For Projected Sdr ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Constant Value = *value*

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
value	real	–

Exponential

Syntax

Source For Projected Sdr ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The source, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
variable	“string”	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Source For Projected Sdr ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Global (Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
global_name	“string”	–
component	integer	0

Lumped_Constant

Syntax

Source For Projected Sdr ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Lumped_Constant Value = *value*

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
value	real	–

Lumped_Exponential

Syntax

Source For Projected Sdr ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Lumped_Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Lumped_Global

Syntax

Source For Projected Sdr ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Lumped_Global (Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
global_name	"string"	–
component	integer	0

Lumped_Nd_Table

Syntax

Source For Projected Sdr ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Lumped_Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Lumped_Polynomial

Syntax

Source For Projected Sdr ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Lumped_Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Lumped_Scalar_String_Function

Syntax

Source For Projected Sdr ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = Lumped_Scalar_String_Function F = *f*

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
f	"string"	–

Lumped_User_Field

Syntax

Source For Projected Sdr ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = Lumped_User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

Lumped_User_Function

Syntax

Source For Projected Sdr ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = Lumped_User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1
toggle	"string"	–

Nd_Table

Syntax

Source For Projected Sdr ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Source For Projected Sdr ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The source, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Source For Projected Sdr ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = Scalar_String_Function F = *f*

Summary

Value from a string function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
f	"string"	–

User_Field

Syntax

Source For Projected Sdr ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name	“string”	–
scaling	real	1
global_var	“string”	–
time_function	“string”	–

User_Function

Syntax

Source For Projected Sdr ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1
toggle	"string"	–

User_Plugin

Syntax

Source For Projected Sdr ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = User_Plugin (Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
Name	"string"	–
plugin_parameters	"string"	–

Projected Tke (

Constant

Syntax

Source For Projected Tke ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Constant Value = *value*

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
value	real	–

Exponential

Syntax

Source For Projected Tke ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The source, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
variable	“string”	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Source For Projected Tke ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = Global (Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
global_name	“string”	–
component	integer	0

Lumped_Constant

Syntax

Source For Projected Tke ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = Lumped_Constant Value = *value*

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
value	real	–

Lumped_Exponential

Syntax

Source For Projected Tke ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Lumped_Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Lumped_Global

Syntax

Source For Projected Tke ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Lumped_Global (Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
global_name	"string"	–
component	integer	0

Lumped_Nd_Table

Syntax

Source For Projected Tke ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Lumped_Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Lumped_Polynomial

Syntax

Source For Projected Tke ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Lumped_Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Lumped_Scalar_String_Function

Syntax

Source For Projected Tke ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = Lumped_Scalar_String_Function F = *f*

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
f	"string"	–

Lumped_User_Field

Syntax

Source For Projected Tke ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = Lumped_User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

Lumped_User_Function

Syntax

Source For Projected Tke ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = Lumped_User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1
toggle	"string"	–

Nd_Table

Syntax

Source For Projected Tke ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Source For Projected Tke ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The source, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Source For Projected Tke ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = Scalar_String_Function F = *f*

Summary

Value from a string function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
f	"string"	–

User_Field

Syntax

Source For Projected Tke ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name	“string”	–
scaling	real	1
global_var	“string”	–
time_function	“string”	–

User_Function

Syntax

Source For Projected Tke ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1
toggle	"string"	–

User_Plugin

Syntax

Source For Projected Tke ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = User_Plugin (Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
Name	"string"	–
plugin_parameters	"string"	–

Solid (

Constant_Vector

Syntax

Source For Solid ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Constant_Vector ([Valuex = *valuex*] | [Valuey = *valuey*] | [Valuez = *valuez*])

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
valuex	real	0
valuey	real	0
valuez	real	0

Hydrostatic

Syntax

Source For Solid ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = Hydrostatic ([Gx = *gx*] | [Gy = *gy*] | [Gz = *gz*] | [Ref_Density = *ref_density*] | [Density_Phase = *density_phase*])

Summary

Hydrostatic source for momentum equation

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
gx	real	0
gy	real	0
gz	real	0
ref_density	real	0
density_phase	“string”	–

Solid_Body_Acceleration

Syntax

Source For Solid ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Solid_Body_Acceleration ()

Summary

solid body acceleration source for solid equation

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–

Spring_Force

Syntax

Source For Solid ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Spring_Force ([Offset_X = *offset_x*] | [Offset_Y = *offset_y*] | [Offset_Z = *offset_z*] | Mult = *mult*)

Summary

Spring force solid body force

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
offset_x	real	0
offset_y	real	0
offset_z	real	0
mult	real	–

User_Field

Syntax

Source For Solid ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a Vector User Field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name	“string”	–
scaling	real	1
global_var	“string”	–
time_function	“string”	–

User_Function

Syntax

Source For Solid ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = User_Function ([Name_X = *name_x*] | [Name_Y = *name_y*] | [Name_Z = *name_z*] | X = *x* | [Multiplier = *multiplier*])

Summary

Value from a Vector User Function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name_x	"string"	–
name_y	"string"	–
name_z	"string"	–
x	"string"	–
multiplier	real	1

Vector_String_Function

Syntax

Source For Solid ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Vector_String_Function (F_X = *f_x* | F_Y = *f_y* | [F_Z = *f_z*])

Summary

Value from a string function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
f_x	"string"	–
f_y	"string"	–
f_z	"string"	–

Solid Body Acceleration (

Constant_Vector

Syntax

Source For Solid Body Acceleration ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Constant_Vector ([Valuex = *valuex*] | [Valuey = *valuey*] | [Valuez = *valuez*])

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
valuex	real	0
valuey	real	0
valuez	real	0

User_Field

Syntax

Source For Solid Body Acceleration ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a Vector User Field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Source For Solid Body Acceleration ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = User_Function ([Name_X = *name_x*] | [Name_Y = *name_y*] | [Name_Z = *name_z*] | X = *x* | [Multiplier = *multiplier*])

Summary

Value from a Vector User Function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name_x	"string"	–
name_y	"string"	–
name_z	"string"	–
x	"string"	–
multiplier	real	1

Vector_String_Function

Syntax

Source For Solid Body Acceleration ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Vector_String_Function (F_X = *f_x* | F_Y = *f_y* | [F_Z = *f_z*])

Summary

Value from a string function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
f_x	"string"	–
f_y	"string"	–
f_z	"string"	–

Sp (

Attenuation

Syntax

Source For Sp ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Attenuation Spn_Order = *SPn_order*

Summary

Black body radiation source for spherical harmonic equation

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
SPn_order	integer	–

Black_Body

Syntax

Source For Sp ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Black_Body Spn_Order = *SPn_order*

Summary

Black body radiation source for spherical harmonic equation

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
SPn_order	integer	–

Constant

Syntax

Source For Sp ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Constant (Spn_Order = *SPn_order* | Value = *value*)

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
SPn_order	integer	–
value	real	–

Exponential

Syntax

Source For Sp ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Exponential (Spn_Order = *SPn_order* | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The source, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
SPn_order	integer	–
variable	“string”	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Source For Sp ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Global (Spn_Order = *SPn_order* | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
SPn_order	integer	–
global_name	"string"	–
component	integer	0

Lumped_Constant

Syntax

Source For Sp ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Lumped_Constant (Spn_Order = *SPn_order* | Value = *value*)

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
SPn_order	integer	–
value	real	–

Lumped_Exponential

Syntax

Source For Sp ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Lumped_Exponential (Spn_Order = *SPn_order* | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
SPn_order	integer	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Lumped_Global

Syntax

Source For Sp ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Lumped_Global (Spn_Order = *SPn_order* | Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
SPn_order	integer	–
global_name	"string"	–
component	integer	0

Lumped_Nd_Table

Syntax

Source For Sp ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Lumped_Nd_Table (Spn_Order = *SPn_order* | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
SPn_order	integer	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Lumped_Polynomial

Syntax

Source For Sp ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Lumped_Polynomial (Spn_Order = *SPn_order* | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
SPn_order	integer	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Lumped_Scalar_String_Function

Syntax

Source For Sp ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Lumped_Scalar_String_Function (Spn_Order = *SPn_order* | F = *f*)

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
SPn_order	integer	–
f	"string"	–

Lumped_User_Field

Syntax

Source For Sp ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Lumped_User_Field (Spn_Order = *SPn_order* | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
SPn_order	integer	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

Lumped_User_Function

Syntax

Source For Sp ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Lumped_User_Function (Spn_Order = *SPn_order* | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
SPn_order	integer	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

Nd_Table

Syntax

Source For Sp ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Nd_Table (Spn_Order = *SPn_order* | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
SPn_order	integer	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Source For Sp ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Polynomial (Spn_Order = *SPn_order* | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The source, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
SPn_order	integer	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Source For Sp ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Scalar_String_Function (Spn_Order = *SPn_order* | F = *f*)

Summary

Value from a string function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
SPn_order	integer	–
f	"string"	–

Scattering

Syntax

Source For Sp ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Scattering Spn_Order = *SPn_order*

Summary

Black body radiation source for spherical harmonic equation

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
SPn_order	integer	–

User_Field

Syntax

Source For Sp ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = User_Field (Spn_Order = *SPn_order* | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
SPn_order	integer	–
name	“string”	–
scaling	real	1
global_var	“string”	–
time_function	“string”	–

User_Function

Syntax

Source For Sp ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = User_Function (Spn_Order = *SPn_order* | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
SPn_order	integer	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Source For Sp ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = User_Plugin (Spn_Order = *SPn_order* | Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
SPn_order	integer	–
Name	"string"	–
plugin_parameters	"string"	–

Species (

Bubble_Growth

Syntax

Source For Species ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Bubble_Growth (Bnd_Species_Name = *bnd_species_name* | [Rmin = *Rmin*] | [Nmin = *nmin*] | [P = *p*])

Summary

Species source associated with bubble growth.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
bnd_species_name	"string"	–
Rmin	real	1e-09
nmin	real	1e-06
p	real	0

Chemeq_Species

Syntax

Source For Species ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Chemeq_Species Model = *model*

Summary

This feature has been deprecated and removed from the code.

Description

warning{This capability has been deprecated and removed}

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
model	“string”	–

Chemistry

Syntax

Source For Species ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Chemistry Mechanism = *mechanism*

Summary

General chemistry species source.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
mechanism	“string”	–

Constant

Syntax

Source For Species ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Constant Value = *value*

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
value	real	–

Curing_Foam_Extent

Syntax

Source For Species ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Curing_Foam_Extent (A1 = *a1* | E1 = *e1* | R = *r* | N = *n* | K1 = *k1* | K2 = *k2* | M = *m* | W = *w* | Beta = *beta* | [T_A = *T_a*] | [T_Buff = *T_buff*] | [A = *a*])

Summary

Curing foam extent source for Species equation

Description

The curing foam extent source term (S) is defined as:

$$S = CA_1 \exp\left(\frac{-E_1}{RT}\right)\left(K_1 + K_2\xi^m\right)\left(1-\xi\right)^n\left(1+Wf\right)^{-\beta}$$

where ξ is the extent of reaction (the current species the source is applied to), and f is the polymerization shift factor.

The optional smoothing coefficient, C is defined as:

$$C = \frac{1}{2}\left(1+\tanh\left(a\left(\frac{2(T-T_a)}{T_{buff}}-1\right)\right)\right)$$

The smoothing coefficient (C) is set to 1 if a, T_a, or T_{buff} are not provided.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
a1	real	–
e1	real	–
r	real	–
n	real	–
k1	real	–
k2	real	–
m	real	–
w	real	–
beta	real	–
T_a	real	-1
T_buff	real	-1
a	real	-1

Curing_Foam_Extent_B

Syntax

Source For Species ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = Curing_Foam_Extent_B (A1 = *a1* | A2 = *a2* | E1 = *e1* | E2 = *e2* | R = *r* | M = *m* | N = *n* | [Tnuc = *tnuc*])

Summary

This model is now deprecated

Description

warning{This capability has been deprecated and removed}

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
a1	real	–
a2	real	–
e1	real	–
e2	real	–
r	real	–
m	real	–
n	real	–
tnuc	real	-1.798e+308

Electrode_Object

Syntax

Source For Species ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Electrode_Object ()

Summary

Electrode object source for species equation to be used with porous electrodes

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–

Electrode_Object_Old

Syntax

Source For Species ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Electrode_Object_Old (F = *f* | Electrodefile = *electrodeFile* | Kmolconversion = *kmolConversion* | [Jconversion = *JConversion*] | [Meterconversion = *meterConversion*] | [Max_Sub_Timesteps = *max_sub_timesteps*] | [Base_Delta = *base_delta*])

Summary

Electrode object source for species equation to be used with porous electrodes

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
f	real	–
electrodeFile	"string"	–
kmolConversion	real	–
JConversion	real	1
meterConversion	real	1
max_sub_timesteps	integer	100
base_delta	real	1e-05

Exponential

Syntax

Source For Species ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The source, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

From_Current_Source

Syntax

Source For Species ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = From_Current_Source (Mult = *mult* | [F = *f*])

Summary

Generating species source from equivalent current source

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
mult	real	–
f	real	9.649e+04

Global

Syntax

Source For Species ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Global (Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
global_name	"string"	–
component	integer	0

Interfacial_Mass_Transport

Syntax

Source For Species ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Interfacial_Mass_Transport (Kh = *Kh* | Gas_Species_Name = *gas_species_name* | Bnd_Species_Name = *bnd_species_name* | [Multiplier = *multiplier*] | [Rmin = *Rmin*] | [Nmin = *nmin*])

Summary

Liquid-gas interfacial source term associated with bubble mass transfer.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
Kh	real	–
gas_species_name	"string"	–
bnd_species_name	"string"	–
multiplier	real	1
Rmin	real	1e-09
nmin	real	1e-06

Ionic_Electroneutrality

Syntax

Source For Species ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Ionic_Electroneutrality ()

Summary

Enforcing electroneutrality for ion species

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–

Lumped_Chemistry

Syntax

Source For Species ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Lumped_Chemistry Mechanism = *mechanism*

Summary

General chemistry species source.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
mechanism	“string”	–

Lumped_Constant

Syntax

Source For Species ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Lumped_Constant Value = *value*

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
value	real	–

Lumped_Electrode_Object

Syntax

Source For Species ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Lumped_Electrode_Object ()

Summary

Electrode object source for species equation to be used with porous electrodes

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–

Lumped_Exponential

Syntax

Source For Species ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Lumped_Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Lumped_Global

Syntax

Source For Species ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Lumped_Global (Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
global_name	"string"	–
component	integer	0

Lumped_Nd_Table

Syntax

Source For Species ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Lumped_Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Lumped_Polynomial

Syntax

Source For Species ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Lumped_Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Lumped_Scalar_String_Function

Syntax

Source For Species ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Lumped_Scalar_String_Function F = *f*

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
f	"string"	–

Lumped_User_Field

Syntax

Source For Species ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Lumped_User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

Lumped_User_Function

Syntax

Source For Species ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Lumped_User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1
toggle	"string"	–

Moving_Mesh

Syntax

Source For Species ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Moving_Mesh ()

Summary

Moving mesh source for conservative form, species. Does not use VRTM CVFEM options

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–

Nd_Table

Syntax

Source For Species ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Source For Species ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The source, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Porous_Decomposition

Syntax

Source For Species ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = Porous_Decomposition (E = *e* | A = *a* | Rho = *rho* | Delta_H = *delta_h* | Coef = *coef* | Pexp = *pexp* | R = *r* | Pref = *Pref* | Tref = *Tref* | Type = *type*)

Summary

Porous decomposition source for Species equation

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
e	real	–
a	real	–
rho	real	–
delta_h	real	–
coef	real	–
pexp	real	–
r	real	–
Pref	real	–
Tref	real	–
type	integer	–

Pressurization_Model

Syntax

Source For Species ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = Pressurization_Model Model_Name = *model_name*

Summary

Gas moles source for use on pressurization model bulk node.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
model_name	“string”	–

Pressurization_Model_Venting

Syntax

Source For Species ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Pressurization_Model_Venting ()

Summary

Automatically created source term to apply venting boundary condition to pressurization model bulk node.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–

Scalar_String_Function

Syntax

Source For Species ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Scalar_String_Function F = *f*

Summary

Value from a string function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
f	“string”	–

User_Field

Syntax

Source For Species ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Source For Species ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1
toggle	"string"	–

User_Plugin

Syntax

Source For Species ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = User_Plugin (Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
Name	"string"	–
plugin_parameters	"string"	–

Species Mass (

Constant

Syntax

Source For Species Mass ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Constant Value = *value*

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
value	real	–

Exponential

Syntax

Source For Species Mass ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The source, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
variable	“string”	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Source For Species Mass ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = Global (Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
global_name	“string”	–
component	integer	0

Lumped_Constant

Syntax

Source For Species Mass ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = Lumped_Constant Value = *value*

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
value	real	–

Lumped_Exponential

Syntax

Source For Species Mass ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Lumped_Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Lumped_Global

Syntax

Source For Species Mass ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Lumped_Global (Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
global_name	"string"	–
component	integer	0

Lumped_Nd_Table

Syntax

Source For Species Mass ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Lumped_Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Lumped_Polynomial

Syntax

Source For Species Mass ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Lumped_Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Lumped_Scalar_String_Function

Syntax

Source For Species Mass ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = Lumped_Scalar_String_Function F = *f*

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
f	"string"	–

Lumped_User_Field

Syntax

Source For Species Mass ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = Lumped_User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

Lumped_User_Function

Syntax

Source For Species Mass ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = Lumped_User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1
toggle	"string"	–

Nd_Table

Syntax

Source For Species Mass ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Source For Species Mass ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The source, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Source For Species Mass ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Scalar_String_Function F = *f*

Summary

Value from a string function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
f	"string"	–

User_Field

Syntax

Source For Species Mass ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name	“string”	–
scaling	real	1
global_var	“string”	–
time_function	“string”	–

User_Function

Syntax

Source For Species Mass ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1
toggle	"string"	–

User_Plugin

Syntax

Source For Species Mass ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = User_Plugin (Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
Name	"string"	–
plugin_parameters	"string"	–

Specific Dissipation Rate (

Constant

Syntax

Source For Specific Dissipation Rate ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Constant Value = *value*

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
value	real	–

Exponential

Syntax

Source For Specific Dissipation Rate ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The source, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
variable	“string”	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Source For Specific Dissipation Rate ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Global (Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
global_name	“string”	–
component	integer	0

Lumped_Constant

Syntax

Source For Specific Dissipation Rate ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Lumped_Constant Value = *value*

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
value	real	–

Lumped_Exponential

Syntax

Source For Specific Dissipation Rate ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Lumped_Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Lumped_Global

Syntax

Source For Specific Dissipation Rate ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Lumped_Global (Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
global_name	"string"	–
component	integer	0

Lumped_Nd_Table

Syntax

Source For Specific Dissipation Rate ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Lumped_Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Lumped_Polynomial

Syntax

Source For Specific Dissipation Rate ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Lumped_Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Lumped_Scalar_String_Function

Syntax

Source For Specific Dissipation Rate ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = Lumped_Scalar_String_Function F = *f*

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
f	"string"	–

Lumped_User_Field

Syntax

Source For Specific Dissipation Rate ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = Lumped_User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

Lumped_User_Function

Syntax

Source For Specific Dissipation Rate ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = Lumped_User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1
toggle	"string"	–

Nd_Table

Syntax

Source For Specific Dissipation Rate ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Source For Specific Dissipation Rate ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The source, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Source For Specific Dissipation Rate ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = Scalar_String_Function F = *f*

Summary

Value from a string function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
f	"string"	–

User_Field

Syntax

Source For Specific Dissipation Rate ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name	“string”	–
scaling	real	1
global_var	“string”	–
time_function	“string”	–

User_Function

Syntax

Source For Specific Dissipation Rate ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Source For Specific Dissipation Rate ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = User_Plugin (Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
Name	“string”	–
plugin_parameters	“string”	–

Suspension (

Constant

Syntax

Source For Suspension ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Constant Value = *value*

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
value	real	–

Exponential

Syntax

Source For Suspension ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The source, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
variable	“string”	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Source For Suspension ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Global (Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
global_name	“string”	–
component	integer	0

Lumped_Constant

Syntax

Source For Suspension ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Lumped_Constant Value = *value*

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
value	real	–

Lumped_Exponential

Syntax

Source For Suspension ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Lumped_Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Lumped_Global

Syntax

Source For Suspension ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Lumped_Global (Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
global_name	"string"	–
component	integer	0

Lumped_Nd_Table

Syntax

Source For Suspension ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Lumped_Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Lumped_Polynomial

Syntax

Source For Suspension ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Lumped_Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Lumped_Scalar_String_Function

Syntax

Source For Suspension ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Lumped_Scalar_String_Function F = *f*

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
f	"string"	–

Lumped_User_Field

Syntax

Source For Suspension ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = Lumped_User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

Lumped_User_Function

Syntax

Source For Suspension ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = Lumped_User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

Nd_Table

Syntax

Source For Suspension ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Source For Suspension ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The source, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Source For Suspension ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = Scalar_String_Function F = *f*

Summary

Value from a string function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
f	"string"	–

User_Field

Syntax

Source For Suspension ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name	“string”	–
scaling	real	1
global_var	“string”	–
time_function	“string”	–

User_Function

Syntax

Source For Suspension ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1
toggle	"string"	–

User_Plugin

Syntax

Source For Suspension ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = User_Plugin (Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
Name	"string"	–
plugin_parameters	"string"	–

Turbulence Dissipation Rate (

Constant

Syntax

Source For Turbulence Dissipation Rate ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Constant Value = *value*

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
value	real	–

Exponential

Syntax

Source For Turbulence Dissipation Rate ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The source, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
variable	“string”	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Source For Turbulence Dissipation Rate ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = Global (Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
global_name	“string”	–
component	integer	0

Lumped_Constant

Syntax

Source For Turbulence Dissipation Rate ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = Lumped_Constant Value = *value*

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
value	real	–

Lumped_Exponential

Syntax

Source For Turbulence Dissipation Rate ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Lumped_Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Lumped_Global

Syntax

Source For Turbulence Dissipation Rate ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Lumped_Global (Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
global_name	"string"	–
component	integer	0

Lumped_Nd_Table

Syntax

Source For Turbulence Dissipation Rate ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Lumped_Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Lumped_Polynomial

Syntax

Source For Turbulence Dissipation Rate ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Lumped_Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Lumped_Scalar_String_Function

Syntax

Source For Turbulence Dissipation Rate ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = Lumped_Scalar_String_Function F = *f*

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
f	"string"	–

Lumped_User_Field

Syntax

Source For Turbulence Dissipation Rate ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = Lumped_User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

Lumped_User_Function

Syntax

Source For Turbulence Dissipation Rate ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = Lumped_User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

Nd_Table

Syntax

Source For Turbulence Dissipation Rate ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Source For Turbulence Dissipation Rate ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The source, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Source For Turbulence Dissipation Rate ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = Scalar_String_Function F = *f*

Summary

Value from a string function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
f	"string"	–

User_Field

Syntax

Source For Turbulence Dissipation Rate ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name	“string”	–
scaling	real	1
global_var	“string”	–
time_function	“string”	–

User_Function

Syntax

Source For Turbulence Dissipation Rate ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Source For Turbulence Dissipation Rate ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = User_Plugin (Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
Name	“string”	–
plugin_parameters	“string”	–

Turbulent Kinetic Energy (

Constant

Syntax

Source For Turbulent Kinetic Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Constant Value = *value*

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
value	real	–

Exponential

Syntax

Source For Turbulent Kinetic Energy ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The source, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
variable	“string”	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Source For Turbulent Kinetic Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Global (Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
global_name	“string”	–
component	integer	0

Lumped_Constant

Syntax

Source For Turbulent Kinetic Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Lumped_Constant Value = *value*

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
value	real	–

Lumped_Exponential

Syntax

Source For Turbulent Kinetic Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Lumped_Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Lumped_Global

Syntax

Source For Turbulent Kinetic Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Lumped_Global (Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
global_name	"string"	–
component	integer	0

Lumped_Nd_Table

Syntax

Source For Turbulent Kinetic Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Lumped_Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Lumped_Polynomial

Syntax

Source For Turbulent Kinetic Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Lumped_Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Lumped_Scalar_String_Function

Syntax

Source For Turbulent Kinetic Energy ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = Lumped_Scalar_String_Function F = *f*

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
f	“string”	–

Lumped_User_Field

Syntax

Source For Turbulent Kinetic Energy ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = Lumped_User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name	“string”	–
scaling	real	1
global_var	“string”	–
time_function	“string”	–

Lumped_User_Function

Syntax

Source For Turbulent Kinetic Energy ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = Lumped_User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

Nd_Table

Syntax

Source For Turbulent Kinetic Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Source For Turbulent Kinetic Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The source, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Source For Turbulent Kinetic Energy ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = Scalar_String_Function F = *f*

Summary

Value from a string function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
f	"string"	–

User_Field

Syntax

Source For Turbulent Kinetic Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name	“string”	–
scaling	real	1
global_var	“string”	–
time_function	“string”	–

User_Function

Syntax

Source For Turbulent Kinetic Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Source For Turbulent Kinetic Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = User_Plugin (Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
Name	“string”	–
plugin_parameters	“string”	–

Voltage (

Charge_Density

Syntax

Source For Voltage ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Charge_Density ()

Summary

charge density source for voltage equation

Description

$S_V = \rho_e$, where ρ_e is the electrical charge density.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–

Constant

Syntax

Source For Voltage ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = Constant Value = *value*

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
value	real	–

Exponential

Syntax

Source For Voltage ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The source, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Source For Voltage ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Global (Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
global_name	"string"	–
component	integer	0

Lumped_Constant

Syntax

Source For Voltage ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Lumped_Constant Value = *value*

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
value	real	–

Lumped_Exponential

Syntax

Source For Voltage ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Lumped_Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
variable	“string”	–
constant	real	0
multiplier	real	1
exponent	real	–

Lumped_Global

Syntax

Source For Voltage ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Lumped_Global (Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
global_name	“string”	–
component	integer	0

Lumped_Nd_Table

Syntax

Source For Voltage ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Lumped_Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Lumped_Polynomial

Syntax

Source For Voltage ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Lumped_Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Lumped_Scalar_String_Function

Syntax

Source For Voltage ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Lumped_Scalar_String_Function F = *f*

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
f	"string"	–

Lumped_User_Field

Syntax

Source For Voltage ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Lumped_User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

Lumped_User_Function

Syntax

Source For Voltage ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Lumped_User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

Nd_Table

Syntax

Source For Voltage ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Source For Voltage ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The source, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Source For Voltage ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Scalar_String_Function F = *f*

Summary

Value from a string function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
f	"string"	–

User_Field

Syntax

Source For Voltage ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Source For Voltage ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1
toggle	"string"	–

User_Plugin

Syntax

Source For Voltage ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = User_Plugin (Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
Name	"string"	–
plugin_parameters	"string"	–

Water Mass Balance (

Constant

Syntax

Source For Water Mass Balance ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Constant Value = *value*

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
value	real	–

Exponential

Syntax

Source For Water Mass Balance ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The source, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
variable	“string”	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Source For Water Mass Balance ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = Global (Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
global_name	“string”	–
component	integer	0

Lumped_Constant

Syntax

Source For Water Mass Balance ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = Lumped_Constant Value = *value*

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
value	real	–

Lumped_Exponential

Syntax

Source For Water Mass Balance ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = Lumped_Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Lumped_Global

Syntax

Source For Water Mass Balance ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = Lumped_Global (Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
global_name	"string"	–
component	integer	0

Lumped_Nd_Table

Syntax

Source For Water Mass Balance ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Lumped_Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Lumped_Polynomial

Syntax

Source For Water Mass Balance ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Lumped_Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Lumped_Scalar_String_Function

Syntax

Source For Water Mass Balance ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = Lumped_Scalar_String_Function F = *f*

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
f	"string"	–

Lumped_User_Field

Syntax

Source For Water Mass Balance ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = Lumped_User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

Lumped_User_Function

Syntax

Source For Water Mass Balance ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = Lumped_User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1
toggle	"string"	–

Nd_Table

Syntax

Source For Water Mass Balance ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Source For Water Mass Balance ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The source, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Source For Water Mass Balance ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = Scalar_String_Function F = *f*

Summary

Value from a string function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
f	"string"	–

User_Field

Syntax

Source For Water Mass Balance ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name	“string”	–
scaling	real	1
global_var	“string”	–
time_function	“string”	–

User_Function

Syntax

Source For Water Mass Balance ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name	“string”	–
x	“string”	–
x_multiplier	real	1
multiplier	real	1
toggle	“string”	–

User_Plugin

Syntax

Source For Water Mass Balance ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = User_Plugin (Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
Name	“string”	–
plugin_parameters	“string”	–

Wetting Phase Mass Balance (

Constant

Syntax

Source For Wetting Phase Mass Balance ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Constant Value = *value*

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
value	real	–

Exponential

Syntax

Source For Wetting Phase Mass Balance ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The source, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
variable	“string”	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Source For Wetting Phase Mass Balance ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = Global (Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
global_name	“string”	–
component	integer	0

Lumped_Constant

Syntax

Source For Wetting Phase Mass Balance ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = Lumped_Constant Value = *value*

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
value	real	–

Lumped_Exponential

Syntax

Source For Wetting Phase Mass Balance ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Lumped_Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Lumped_Global

Syntax

Source For Wetting Phase Mass Balance ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Lumped_Global (Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
global_name	"string"	–
component	integer	0

Lumped_Nd_Table

Syntax

Source For Wetting Phase Mass Balance ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Lumped_Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Lumped_Polynomial

Syntax

Source For Wetting Phase Mass Balance ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Lumped_Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Lumped_Scalar_String_Function

Syntax

Source For Wetting Phase Mass Balance ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = Lumped_Scalar_String_Function F = *f*

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
f	"string"	–

Lumped_User_Field

Syntax

Source For Wetting Phase Mass Balance ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = Lumped_User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

Lumped_User_Function

Syntax

Source For Wetting Phase Mass Balance ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = Lumped_User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1
toggle	"string"	–

Nd_Table

Syntax

Source For Wetting Phase Mass Balance ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Source For Wetting Phase Mass Balance ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The source, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Source For Wetting Phase Mass Balance ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *MeshPartIdentifier* = Scalar_String_Function F = *f*

Summary

Value from a string function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
f	"string"	–

User_Field

Syntax

Source For Wetting Phase Mass Balance ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name	“string”	–
scaling	real	1
global_var	“string”	–
time_function	“string”	–

User_Function

Syntax

Source For Wetting Phase Mass Balance ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1
toggle	"string"	–

User_Plugin

Syntax

Source For Wetting Phase Mass Balance ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} MeshPartIdentifier* = User_Plugin (Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
MeshPartIdentifier	string	–
Name	"string"	–
plugin_parameters	"string"	–

Legacy Sources

Volume Heating

Scope

Aria Region, Equation System, Explicit Equation System, Root Finder Equation System

Summary

Defines a volumetric heat source. Exactly one heating type specification is allowed.

Description

The volumetric heating must be applied to at least one volume. If a user subroutine is specified, then real and integer data may be declared that will be local in scope to this instance of the volume heating.

```
begin Volume Heating Name

   Add Volume VolumeList...

   Chemeq {=} Model

   Element Subroutine {=} Name

   Element Variable {=} Name

   Field Scaling {=} Scale_factor

   Integer Data Values...

   Integrated Power Output VariableName

   Joule Heating

   Nodal Variable {=} Name

   Real Data Values...

   Scaling Time Function {=} Function

   Scaling With Global Variable {=} name

   Source Fortran Subroutine {=} Name

   Temperature Function {=} Function

   Time Function {=} Function

   Use Data Block Name

   Use Toggle Block ToggleName [ {on} ElementBlockList...  ]

   Use Verdi
```

```
   Value {=} vhs

   Verdi Qdot Scaling {=} Scale_factor For Keyval...

end Volume Heating Name
```

Line Commands

Add Volume

Syntax

Add Volume *VolumeList...*

Summary

Specifies the mesh volume to which this code feature is to be applied. More than one volume name may be specified, as a space-delimited list.

Parameter	Value	Default
VolumeList	string...	–

Chemeq

Syntax

Chemeq *{=} Model*

Summary

Specifies that volume heating is due to chemical reaction. Here MODEL pertains to the paired CHEMEQ MODEL and CHEMEQ SOLVER command blocks that define the chemical system and how it can evolve. Chemical heating source contributions are further controlled through parameters in the CHEMEQ SOLVER command block.

Parameter	Value	Default
{=}	{= | are | is}	–
Model	string	–

Element Subroutine

Syntax

Element Subroutine *{=} Name*

Summary

Specifies that the named user-defined subroutine be used. At most one subroutine name may be specified for a given quantity. This subroutine must conform to the "element signature" argument list type.

Parameter	Value	Default
{=}	{= | are | is}	–
Name	string	–

Element Variable

Syntax

Element Variable *{=}* *Name*

Summary

Specifies the name of an element variable in which the volumetric heat source is stored. The units of this term are power per unit volume. Currently, this must be constant in space over each finite element.

Parameter	Value	Default
{=}	{= | are | is}	–
Name	string	–

Field Scaling

Syntax

Field Scaling *{=}* *Scale_factor*

Summary

This command is only relevant when the NODAL VARIABLE or ELEMENT VARIABLE command line option is present. The command enables scaling of the Field variable for the element blocks within this Volume Heat command block.

Parameter	Value	Default
{=}	{= | are | is}	–
Scale_factor	real	–

Integer Data

Syntax

Integer Data *Values. . .*

Summary

List of integer data values to be used by the FORTRAN user subroutine. Copies of these values are provided to the subroutine hence changes to these values within the subroutine are not saved.

Parameter	Value	Default
Values	integer. . .	–

Integrated Power Output

Syntax

Integrated Power Output *VariableName*

Summary

Calculate the total power associated with this flux boundary condition.

Description

This line command specifies that, as a postprocess, the normal flux associated with this boundary condition be integrated over the surface to obtain the total power which is then stored into a global variable named "VariableName". This global variable may then be output to history files, or accessed in user subroutines, etc.

Parameter	Value	Default
VariableName	string	–

Joule Heating

Syntax

Joule Heating

Summary

Specifies that Joule heating will be applied. This model requires that either a VOLTAGE or CURRENT Equation be defined on the specified element block(s). Additionally the corresponding electrical conductivity must be defined.

Nodal Variable

Syntax

Nodal Variable *{=}* *Name*

Summary

Specifies the name of a nodal variable in which the volumetric heat source is stored. The units of this term are power per unit volume.

Description

The nodal variable could be user defined, which could originate from a transfer in the case of coupled mechanics, or from some user subroutine. The nodal values are interpolated to the Gauss points and then integrated over each element.

Parameter	Value	Default
{=}	{= | are | is}	–
Name	string	–

Real Data

Syntax

Real Data *Values. . .*

Summary

List of real data values to be used by the FORTRAN user subroutine. Copies of these values are provided to the subroutine hence changes to these values within the subroutine are not saved.

Parameter	Value	Default
Values	real. . .	–

Scaling Time Function

Syntax

Scaling Time Function *{=}* *Function*

Summary

This command is only relevant when the NODAL VARIABLE or ELEMENT VARIABLE command line option is present. This command enables time function scaling of the field variable for the element blocks within this Volume Heat command block.

Parameter	Value	Default
{=}	{= | are | is}	–
Function	string	–

Scaling With Global Variable

Syntax

Scaling With Global Variable *{=}* *name*

Summary

This command is only relevant when the NODE VARIABLE or ELEMENT VARIABLE command line option is present. The command enables scaling of the source Field by a global variable for the surfaces within the Volume Heat command block. The scalar global variable itself must be defined at the Region scope as type double with length 1.

Parameter	Value	Default
{=}	{= | are | is}	–
name	string	–

Source Fortran Subroutine

Syntax

Source Fortran Subroutine *{=}* *Name*

Summary

Specify the name of a FORTRAN user subroutine that will be used to calculate the volumetric source.

Parameter	Value	Default
{=}	{= | are | is}	–
Name	string	–

Temperature Function

Syntax

Temperature Function *{=}* *Function*

Summary

Specifies a temperature dependent function name for a volumetric heat source to be applied in the named volume.

Parameter	Value	Default
{=}	{= | are | is}	–
Function	string	–

Time Function

Syntax

Time Function *{=}* *Function*

Summary

Specifies a time function dependent name for a volumetric heat source to be applied in the named volume.

Parameter	Value	Default
{=}	{= | are | is}	–
Function	string	–

Use Data Block

Syntax

Use Data Block *Name*

Summary

Reference to predefined data to be used by the user subroutine. These values may be changed by the user subroutine.

Parameter	Value	Default
Name	string	–

Use Toggle Block

Syntax

Use Toggle Block *ToggleName* [*{on}* *ElementBlockList. . .*]

Summary

Specification for toggling entities in the computational model based on Toggle Block parameters. When used at the region level, the list of element blocks to be toggled must be provided. Otherwise a listing of entities is not needed as the Toggle Block will be associated with the command line or the enclosing command block.

Parameter	Value	Default
ToggleName	string	–

Use Verdi

Syntax

Use Verdi

Summary

Specifies that the external Goodyear code VERDI should be used to calculate the volumetric heating source term. The value is assumed to be constant in space over each finite element. WARNINGS: There are some caveats: the code will abort (1) with multi-processor runs, since Verdi is inherently serial; (2) with 3-d simulations, since Verdi expects a 2D heat problem to be solved; (3) whenever mesh adaptivity is in use, since Verdi uses three different meshes in its work that must remain very closely related; and (4) whenever Verdi is called from an executable other than eagle (including Calore), since only eagle links against Verdi.

Value

Syntax

Value *{=}* *vhs*

Summary

Specifies a constant value for a volumetric heat source to be applied in the named volume.

Parameter	Value	Default
{=}	{= | are | is}	–
vhs	real	–

Verdi Qdot Scaling

Syntax

Verdi Qdot Scaling *{=}* *Scale_factor* For *Keyval. . .*

Summary

This command is only relevant when the USE VERDI command line option is present. The command enables scaling of the element Qdot scale factor for a collection of element blocks. This command can appear within the command block several times with different scaling factors for each set of elements. Qdot scaling for element blocks not appearing in a VERDI QDOT SCALING command line is set to unity.

Parameter	Value	Default
{=}	{= | are | is}	–
Scale_factor	real	–
Keyval	string. . .	–

Aria Region General Commands

7.18 Aria Region

Aria Region

Scope

Procedure

Summary

Contains the commands needed to execute an analysis on this region.

```
begin Aria Region RegionName

   Add Nonlinear Control Control_name [ Control_parameters...  ]

   Add Utility Utility_name [ Utility_parameters...  ]

   Advection Velocity For EquationName {=} AdvectionVelocityModel [␣
↪Parameters...  ]

   Anneal Mesh On Startup

   Axisymmetry Axis {=} {x | y}

   Check Conservation Of variable

   Combine Rebalance Groups {=} CombinedRebalanceGroups...

   Constrain Dof DofName On {cdfem_edge_nodes | cdfem_internal_nodes |␣
↪cdfem_parent_nodes} [ On SurfaceName  ]
```

```
  Create Element Field Field_name Of Type {asym_tensor_03 | complex |␣
↪full_tensor_22 | full_tensor_36 | integer | long_integer | matrix_22␣
↪| matrix_33 | real | sym_tensor_21 | sym_tensor_31 | sym_tensor_33 |␣
↪unsigned_integer | unsigned_integer_64 | vector_2d | vector_3d} And␣
↪Dimension Dimension [ On Part_list...  ]

  Create Face Field Field_name Of Type {asym_tensor_03 | complex |␣
↪full_tensor_22 | full_tensor_36 | integer | long_integer | matrix_22␣
↪| matrix_33 | real | sym_tensor_21 | sym_tensor_31 | sym_tensor_33 |␣
↪unsigned_integer | unsigned_integer_64 | vector_2d | vector_3d} And␣
↪Dimension Dimension [ On Part_list...  ]

  Create Nodal Field Field_name Of Type {asym_tensor_03 | complex |␣
↪full_tensor_22 | full_tensor_36 | integer | long_integer | matrix_22␣
↪| matrix_33 | real | sym_tensor_21 | sym_tensor_31 | sym_tensor_33 |␣
↪unsigned_integer | unsigned_integer_64 | vector_2d | vector_3d} And␣
↪Dimension Dimension [ On Part_list...  ]

  Display Truncation Error For Solution_dof [ With HowMany Entries  ]

  Enable Rebalance [ With Threshold {=} Threshold  ]

  Eq EquationName For AssociatedDoF {on} MeshPartIdentifier Using␣
↪ElementType With Terms...

  Eq Ampere For Associated DoF ([{of} SpeciesName]  | [{in}␣
↪MaterialPhaseName]  | [{ls} {a | b | c}]) {@ | at | for | on | over}␣
↪Mesh Extent Name Using Element Type With Terms...

  Eq Artificial Compressibility For Associated DoF ([{of} SpeciesName] ␣
↪| [{in} MaterialPhaseName]  | [{ls} {a | b | c}]) {@ | at | for | on |␣
↪over} Mesh Extent Name Using Element Type With Terms...

  Eq Brinkman Momentum For Associated DoF ([{of} SpeciesName]  | [{in}
↪ MaterialPhaseName]  | [{ls} {a | b | c}]) {@ | at | for | on | over}␣
↪Mesh Extent Name Using Element Type With Terms...

  Eq Charge Density For Associated DoF ([{of} SpeciesName]  | [{in}␣
↪MaterialPhaseName]  | [{ls} {a | b | c}]) {@ | at | for | on | over}␣
↪Mesh Extent Name Using Element Type With Terms...

  Eq Continuity For Associated DoF ([{of} SpeciesName]  | [{in}␣
↪MaterialPhaseName]  | [{ls} {a | b | c}]) {@ | at | for | on | over}␣
↪Mesh Extent Name Using Element Type With Terms...
```

```
  Eq Current For Associated DoF ([{of} SpeciesName]  | [{in}␣
↪MaterialPhaseName]  | [{ls} {a | b | c}]) {@ | at | for | on | over}␣
↪Mesh Extent Name Using Element Type With Terms...

  Eq Cvfem Continuity For Associated DoF ([{of} SpeciesName]  | [{in}␣
↪MaterialPhaseName]  | [{ls} {a | b | c}]) {@ | at | for | on | over}␣
↪Mesh Extent Name Using Element Type With Terms...

  Eq Cvfem Dispersed Continuity For Associated DoF ([{of} SpeciesName] ␣
↪| [{in} MaterialPhaseName]  | [{ls} {a | b | c}]) {@ | at | for | on |␣
↪over} Mesh Extent Name Using Element Type With Terms...

  Eq Cvfem Dispersed Momentum For Associated DoF ([{of} SpeciesName] ␣
↪| [{in} MaterialPhaseName]  | [{ls} {a | b | c}]) {@ | at | for | on |␣
↪over} Mesh Extent Name Using Element Type With Terms...

  Eq Cvfem Div Projection For Associated DoF ([{of} SpeciesName]  | [{in}
↪ MaterialPhaseName]  | [{ls} {a | b | c}]) {@ | at | for | on | over}␣
↪Mesh Extent Name Using Element Type With Terms...

  Eq Cvfem Energy For Associated DoF ([{of} SpeciesName]  | [{in}␣
↪MaterialPhaseName]  | [{ls} {a | b | c}]) {@ | at | for | on | over}␣
↪Mesh Extent Name Using Element Type With Terms...

  Eq Cvfem Lagged Muscl Projection For Associated DoF ([{of}␣
↪SpeciesName]  | [{in} MaterialPhaseName]  | [{ls} {a | b | c}]) {@ |␣
↪at | for | on | over} Mesh Extent Name Using Element Type With Terms...

  Eq Cvfem Lagged Projection For Associated DoF ([{of} SpeciesName]  |␣
↪[{in} MaterialPhaseName]  | [{ls} {a | b | c}]) {@ | at | for | on |␣
↪over} Mesh Extent Name Using Element Type With Terms...

  Eq Cvfem Level Set For Associated DoF ([{of} SpeciesName]  | [{in}␣
↪MaterialPhaseName]  | [{ls} {a | b | c}]) {@ | at | for | on | over}␣
↪Mesh Extent Name Using Element Type With Terms...

  Eq Cvfem Lumped Div Projection For Associated DoF ([{of} SpeciesName] ␣
↪| [{in} MaterialPhaseName]  | [{ls} {a | b | c}]) {@ | at | for | on |␣
↪over} Mesh Extent Name Using Element Type With Terms...

  Eq Cvfem Lumped Muscl Projection For Associated DoF ([{of}␣
↪SpeciesName]  | [{in} MaterialPhaseName]  | [{ls} {a | b | c}]) {@ |␣
↪at | for | on | over} Mesh Extent Name Using Element Type With Terms...
```

Eq Cvfem Lumped Projection For *Associated DoF* ([*{of} SpeciesName*] |␣
↪[*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on |␣
↪over} Mesh Extent Name* Using *Element Type* With *Terms...*

Eq Cvfem Mass Fraction For *Associated DoF* ([*{of} SpeciesName*] | [*{in}
↪ MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over}␣
↪Mesh Extent Name* Using *Element Type* With *Terms...*

Eq Cvfem Mesh For *Associated DoF* ([*{of} SpeciesName*] | [*{in}␣
↪MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over}␣
↪Mesh Extent Name* Using *Element Type* With *Terms...*

Eq Cvfem Mixture Fraction For *Associated DoF* ([*{of} SpeciesName*] |␣
↪[*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on |␣
↪over} Mesh Extent Name* Using *Element Type* With *Terms...*

Eq Cvfem Momentum For *Associated DoF* ([*{of} SpeciesName*] | [*{in}␣
↪MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over}␣
↪Mesh Extent Name* Using *Element Type* With *Terms...*

Eq Cvfem Muscl Projection For *Associated DoF* ([*{of} SpeciesName*] |␣
↪[*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on |␣
↪over} Mesh Extent Name* Using *Element Type* With *Terms...*

Eq Cvfem Projection For *Associated DoF* ([*{of} SpeciesName*] | [*{in}␣
↪MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over}␣
↪Mesh Extent Name* Using *Element Type* With *Terms...*

Eq Cvfem Phase Balance For *Associated DoF* ([*{of} SpeciesName*] | [*{in}
↪ MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over}␣
↪Mesh Extent Name* Using *Element Type* With *Terms...*

Eq Cvfem Quasi Implicit Muscl Projection For *Associated DoF* ([*{of}␣
↪SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at␣
↪| for | on | over} Mesh Extent Name* Using *Element Type* With *Terms...*

Eq Cvfem Solvent Continuity For *Associated DoF* ([*{of} SpeciesName*] ␣
↪| [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on |␣
↪over} Mesh Extent Name* Using *Element Type* With *Terms...*

Eq Cvfem Solvent Momentum For *Associated DoF* ([*{of} SpeciesName*] |␣
↪[*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on |␣
↪over} Mesh Extent Name* Using *Element Type* With *Terms...*

Eq Cvfem Specific Dissipation Rate For *Associated DoF* ([*{of}␣

↪*SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ |*␣
↪*at | for | on | over} Mesh Extent Name* Using *Element Type* With *Terms...*

Eq Cvfem Turbulent Kinetic Energy For *Associated DoF* ([*{of}*␣
↪*SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ |*␣
↪*at | for | on | over} Mesh Extent Name* Using *Element Type* With *Terms...*

Eq Cvfem Turbulence Dissipation Rate For *Associated DoF* ([*{of}*␣
↪*SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ |*␣
↪*at | for | on | over} Mesh Extent Name* Using *Element Type* With *Terms...*

Eq Cvfem Velocity Pressure Projection For *Associated DoF* ([*{of}*␣
↪*SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ |*␣
↪*at | for | on | over} Mesh Extent Name* Using *Element Type* With *Terms...*

Eq Darcy Momentum For *Associated DoF* ([*{of} SpeciesName*] | [*{in}*␣
↪*MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over}*␣
↪*Mesh Extent Name* Using *Element Type* With *Terms...*

Eq Div Projection For *Associated DoF* ([*{of} SpeciesName*] | [*{in}*␣
↪*MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over}*␣
↪*Mesh Extent Name* Using *Element Type* With *Terms...*

Eq Energy For *Associated DoF* ([*{of} SpeciesName*] | [*{in}*␣
↪*MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over}*␣
↪*Mesh Extent Name* Using *Element Type* With *Terms...*

Eq Enthalpy For *Associated DoF* ([*{of} SpeciesName*] | [*{in}*␣
↪*MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over}*␣
↪*Mesh Extent Name* Using *Element Type* With *Terms...*

Eq Extension Speed For *Associated DoF* ([*{of} SpeciesName*] | [*{in}*␣
↪*MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over}*␣
↪*Mesh Extent Name* Using *Element Type* With *Terms...*

Eq Hdiff For *Associated DoF* ([*{of} SpeciesName*] | [*{in}*␣
↪*MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over}*␣
↪*Mesh Extent Name* Using *Element Type* With *Terms...*

Eq Lagged Projection For *Associated DoF* ([*{of} SpeciesName*] | [*{in}*
↪ *MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over}*␣
↪*Mesh Extent Name* Using *Element Type* With *Terms...*

Eq Level Set For *Associated DoF* ([*{of} SpeciesName*] | [*{in}*␣
↪*MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over}*␣

↪*Mesh Extent Name* Using *Element Type* With *Terms...*

Eq Lubrication For *Associated DoF* ([*{of} SpeciesName*] | [*{in}*␣
↪*MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over}*␣
↪*Mesh Extent Name* Using *Element Type* With *Terms...*

Eq Lubrication Height For *Associated DoF* ([*{of} SpeciesName*] | [*{in}*
↪ *MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over}*␣
↪*Mesh Extent Name* Using *Element Type* With *Terms...*

Eq Lumped Div Projection For *Associated DoF* ([*{of} SpeciesName*] | [
↪*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on |*␣
↪*over} Mesh Extent Name* Using *Element Type* With *Terms...*

Eq Lumped Projection For *Associated DoF* ([*{of} SpeciesName*] | [*{in}*
↪ *MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over}*␣
↪*Mesh Extent Name* Using *Element Type* With *Terms...*

Eq Magnetic Continuity For *Associated DoF* ([*{of} SpeciesName*] | [*{in}*
↪ *MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over}*␣
↪*Mesh Extent Name* Using *Element Type* With *Terms...*

Eq Magnetic Flux For *Associated DoF* ([*{of} SpeciesName*] | [*{in}*␣
↪*MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over}*␣
↪*Mesh Extent Name* Using *Element Type* With *Terms...*

Eq Magnetic Flux Intensity For *Associated DoF* ([*{of} SpeciesName*] |␣
↪[*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on |*␣
↪*over} Mesh Extent Name* Using *Element Type* With *Terms...*

Eq Mass Balance For *Associated DoF* ([*{of} SpeciesName*] | [*{in}*␣
↪*MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over}*␣
↪*Mesh Extent Name* Using *Element Type* With *Terms...*

Eq Mesh For *Associated DoF* ([*{of} SpeciesName*] | [*{in}*␣
↪*MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over}*␣
↪*Mesh Extent Name* Using *Element Type* With *Terms...*

Eq Mixture Fraction For *Associated DoF* ([*{of} SpeciesName*] | [*{in}*␣
↪*MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over}*␣
↪*Mesh Extent Name* Using *Element Type* With *Terms...*

Eq Momentum For *Associated DoF* ([*{of} SpeciesName*] | [*{in}*␣
↪*MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over}*␣
↪*Mesh Extent Name* Using *Element Type* With *Terms...*

Eq Bar Momentum For *Associated DoF* ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* Using *Element Type* With *Terms...*

Eq Neutron For *Associated DoF* ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* Using *Element Type* With *Terms...*

Eq Porous Enthalpy For *Associated DoF* ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* Using *Element Type* With *Terms...*

Eq Porous Species For *Associated DoF* ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* Using *Element Type* With *Terms...*

Eq Potential For *Associated DoF* ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* Using *Element Type* With *Terms...*

Eq Precursor Conservation For *Associated DoF* ([*{of}* *SpeciesName*] |[*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on |over}* *Mesh Extent Name* Using *Element Type* With *Terms...*

Eq Projection For *Associated DoF* ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* Using *Element Type* With *Terms...*

Eq Quasi Implicit Projection For *Associated DoF* ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on |over}* *Mesh Extent Name* Using *Element Type* With *Terms...*

Eq Radiation Transport For *Associated DoF* ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* Using *Element Type* With *Terms...*

Eq Semilagrangian Level Set For *Associated DoF* ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on |over}* *Mesh Extent Name* Using *Element Type* With *Terms...*

Eq Solid For *Associated DoF* ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* Using *Element Type* With *Terms...*

```
  Eq Sp For Associated DoF ([{of} SpeciesName]  | [{in}␣
↪MaterialPhaseName]  | [{ls} {a | b | c}]) {@ | at | for | on | over}␣
↪Mesh Extent Name Using Element Type With Terms...

  Eq Species For Associated DoF ([{of} SpeciesName]  | [{in}␣
↪MaterialPhaseName]  | [{ls} {a | b | c}]) {@ | at | for | on | over}␣
↪Mesh Extent Name Using Element Type With Terms...

  Eq Suspension For Associated DoF ([{of} SpeciesName]  | [{in}␣
↪MaterialPhaseName]  | [{ls} {a | b | c}]) {@ | at | for | on | over}␣
↪Mesh Extent Name Using Element Type With Terms...

  Eq User Scalar For Associated DoF ([{of} SpeciesName]  | [{in}␣
↪MaterialPhaseName]  | [{ls} {a | b | c}]) {@ | at | for | on | over}␣
↪Mesh Extent Name Using Element Type With Terms...

  Eq User Tensor For Associated DoF ([{of} SpeciesName]  | [{in}␣
↪MaterialPhaseName]  | [{ls} {a | b | c}]) {@ | at | for | on | over}␣
↪Mesh Extent Name Using Element Type With Terms...

  Eq User Vector For Associated DoF ([{of} SpeciesName]  | [{in}␣
↪MaterialPhaseName]  | [{ls} {a | b | c}]) {@ | at | for | on | over}␣
↪Mesh Extent Name Using Element Type With Terms...

  Eq Voltage For Associated DoF ([{of} SpeciesName]  | [{in}␣
↪MaterialPhaseName]  | [{ls} {a | b | c}]) {@ | at | for | on | over}␣
↪Mesh Extent Name Using Element Type With Terms...

  Eq Scaled_Sp Order For AssociatedDoF {on} MeshPartIdentifier Using␣
↪ElementType [ With Src  ]

  Global Variables Excluded From Log File {=} variable_names...

  Integration Rule For BlockName {=} Order

  List Mesh Groups [ Name...  ]

  Maximum Number Of Nodal Rebalance Iterations {=} Count

  Maximum Number Of Rebalances {=} Count

  Mesh Group name {=} Mesh_extent...

  Minimum Nonlinear Iterations {=} Np
```

```
  No Viewfactor Surface Randomize

  Nonlinear Relative Correction Tolerance {=} Pt

  Nonlinear Solver {=} Solver

  Only Compute Viewfactors

  Phase Change Relaxation Factor {=} Prf

  Point Source For EquationName ([{of} SpeciesName]  | [{in}␣
↪MaterialPhaseName]) {on} MeshPartIdentifier Value {=} PointSourceValue␣
↪X {=} Xcoord [ Y {=} Ycoord Z {=} Zcoord  ]

  Post Process {integral | normalized} VariableName On IoPartList... As␣
↪OutputName [ Using WeightList... Weight  ]

  Post Process Flux variableName On ioPartList... As outputName [␣
↪Restricted To restrictPartList...  ]

  Post Process Nodal {integral | normalized} VariableName On IoPartList..
↪. As OutputName [ Using WeightList... Weight  ]

  Predictor Fields {=} Field_names...

  Preprocess Enclosure Topology [ Options...  ]

  Preprocess Enclosures [ Options...  ]

  Pressure Stabilization Is {no_stabilization | pgps_dt | ppps_constant␣
↪| pspg_constant | pspg_global | pspg_local | pspg_projected_shakib |␣
↪pspg_projected_unsteady_shakib | pspg_shakib | pspg_unsteady | pspg_
↪unsteady_shakib | pspp_constant | pspp_off_interface | pspp_transient}␣
↪With Scaling {=} ScalingParam

  Rebalance After Adaptivity

  Rebalance Load Measure Type {=} MeasureType

  Rebalance Time Step Frequency {=} Frequency

  Residual Weight For dof_name {=} weight

  Root Finder Attempt Flux Solve Before Bisection {=} {false | no | off␣
↪| on | true | yes}
```

```
  Root Finder Controls Bc For Equation EquationName {on}␣
↪MeshPartIdentifier Initialvalue {=} InitialValue

  Root Finder Correlation Between Flux And Dirichlet Is Positive {=}
↪{false | no | off | on | true | yes}

  Root Finder Dirichlet Tolerance Is DirichletTolerance

  Root Finder Flux Tolerance Is FluxTolerance

  Root Finder Target Flux Expressions Are FluxExpressionNames...

  Root Finder Target Flux Value Is FluxTarget

  Save Coeff {all_numerical_methods | ccfvm | cvfem | galerkin_
↪fem} Expression ExpressionName ([{of} SpeciesName]  | [{in}␣
↪MaterialPhaseName]) As FieldName [ On Parts partList...  ]

  Save Ip {ccfvm | cvfem | galerkin_fem | scs_cvfem | scv_
↪cvfem} Expression ExpressionName ([{of} SpeciesName]  | [{in}␣
↪MaterialPhaseName]) As FieldName On Parts partList...

  Save Residuals {=} mode

  Segregated Global Minimum Convergence Rate {=} MinRate [ Number Of␣
↪Steps {=} MinSteps  ]

  Segregated Global Nonlinear Initial Residual Tolerance {=} Pt

  Segregated Global Nonlinear Number Of Steps {=} Number_of_steps [ Fail␣
↪Step {=} {false | no | off | on | true | yes}  ]

  Segregated Global Nonlinear Relative Residual Tolerance {=} Pt

  Segregated Global Nonlinear Residual Tolerance {=} Pt [ Minimum Steps
↪{=} minStepsValue  ]

  Segregated Global Nonlinear Target Number Of Steps Per Time Step {=}␣
↪target_iterations

  Stk Bucket Capacity {=} value

  Stk Initial Bucket Capacity {=} value
```

```
  Stk Maximum Bucket Capacity {=} value

  Switch Secondary Solver On Recoverable Linear Solve Failure

  Species Names...

  Thermal Strain Reference Temperature {=} Value

  Track Irreversible Phase Change Name Variable {< | > | cdfem␣
↪inequality criterion undefined} Value

  Use Coeff Expression ExpressionName ([{of} SpeciesName]  | [{in}␣
↪MaterialPhaseName]) From FieldName

  Use Dash Enclosures [ type tol {=} Params...  ]

  Use Data Block Name

  Use Ip Expression ExpressionName ([{of} SpeciesName]  | [{in}␣
↪MaterialPhaseName]) From FieldName

  Use Percept For Adaptivity

  User Field REAL_or_INTEGER NODE_or_FACE_or_ELEMENT SCALAR_or_VECTOR_
↪or_TENSOR Variable_name On Mesh_extent ([Smooth nsmooth Iterations]  |␣
↪[Value {=} Magnitude...])

  Use Finite Element Model ModelName [ Model Coordinates Are Nodal_
↪variable_name  ]

  Vector Point Source For EquationName ([{of} SpeciesName]  | [{in}␣
↪MaterialPhaseName]) {on} MeshPartIdentifier V_X {=} X_PointSourceValue␣
↪V_Y {=} Y_PointSourceValue V_Z {=} Z_PointSourceValue At X {=} Xcoord Y
↪{=} Ycoord Z {=} Zcoord

  Define Global Scalar variable-name... As {int | real} Operation {max␣
↪| min | sum} ([Initial Value initial-value]  | [Reset On {nonlinear␣
↪iteration | nonlinear step}])

  Initial Uniform Refinement For Num Iterations

  begin Bounding Surface InterfaceName
  end

  begin Closed Surface Volume ModelName
```

```
end

begin Conditional Nodelist Value
end

begin Directed Energy Model Model Name
end

begin Data Probe probeName
end

begin Electrochemical Loss Postprocessor LossPostProcName
end

begin Equation System Name
end

begin Explicit Equation System Name
end

begin Heartbeat Output Label
end

begin Ode Solver Parameters SolverName
end

begin Raytrace Source Value
end

begin Restart Data Label
end

begin Root Finder Equation System Name
end

begin Solution Options OptionsName
end

begin Submodel SubModel_name
end

begin User Variable Name
end

begin Data Block Block Name
```

end

end Aria Region *RegionName*

Line Commands

Add Nonlinear Control

Syntax

Add Nonlinear Control *Control_name* [*Control_parameters. . .*]

Summary

Add the named nonlinear control to the multilevel nonlinear solver. Controls are added in the order they appear in the input file and each new control is added at the next higher level. Control parameters can be supplied in the form of key=value pairs.

Parameter	Value	Default
Control_name	string	–

Add Utility

Syntax

Add Utility *Utility_name* [*Utility_parameters. . .*]

Summary

Add the named utility to the nonlinear solver which executes at the specified time. Utility parameters can be supplied in the form of key=value pairs.

Parameter	Value	Default
Utility_name	string	–

Advection Velocity For

Syntax

Advection Velocity For *EquationName {=} AdvectionVelocityModel* [*Parameters. . .*]

Summary

Causes the specified equation to use the specified advection velocity. If this command is not given, the standard advection velocity is used: fluid velocity minus the mesh velocity.

Parameter	Value	Default
EquationName	string	–
{=}	{= | are | is}	–
AdvectionVelocityModel	string	–

Anneal Mesh On Startup

Syntax

Anneal Mesh On Startup

Summary

With the command in the Region block Aria will search for a displacements field and, if found, will apply the displacements to the model and physical coordinates.

Axisymmetry Axis

Syntax

Axisymmetry Axis *{=} {x | y}*

Summary

When the COORDINATE SYSTEM = AXISYMMETRIC command appears within the Finite Element Model command block one can select at the Region level the coordinate axis about which the 2D axisymmetric sweep will occur. Note that there can be no overlap between the model and the axis of rotation.

Parameter	Value	Default
{=}	{= | are | is}	–
AxisType	{x | y}	Y

Check Conservation

Syntax

Check Conservation Of *variable*

Summary

Enable conservation checking for a specified variable in this equation system.

Parameter	Value	Default
variable	string	ENERGY

Combine Rebalance Groups

Syntax

Combine Rebalance Groups *{=} CombinedRebalanceGroups. . .*

Summary

Specifies a list of rebalance groups to combine into the DEFAULT rebalance group for the equation system. By default, each equation system will be split into the following four rebalance groups: DEFAULT, ASSEMBLY, LINEAR_SOLVE, and CHEMEQ_UTIL. Only the ASSEMBLY and CHEMEQ_UTIL rebalance groups can be combined into the DEFAULT rebalance group. The LINEAR_SOLVE rebalance group uses a non-time based measure of the work load and so can not be combined into the DEFAULT rebalance group. Note that combining rebalance groups may hide load imbalances in the equation system.

Parameter	Value	Default
{=}	{= | are | is}	–
CombinedRebalanceGroups	string. . .	–

Constrain Dof

Syntax

Constrain Dof *DofName* On *{cdfem_edge_nodes | cdfem_internal_nodes | cdfem_parent_nodes}* [On *SurfaceName*]

Summary

Constrain a degree of freedom in CDFEM subelements.

Parameter	Value	Default
DofName	string	–
ConstraintLocationName	{cdfem_edge_nodes | cdfem_internal_nodes | cdfem_parent_nodes}	–

Create Element Field

Syntax

Create Element Field *Field_name* Of Type *{asym_tensor_03 | complex | full_tensor_22 | full_tensor_36 | integer | long_integer | matrix_22 | matrix_33 | real | sym_tensor_21 | sym_tensor_31 | sym_tensor_33 | unsigned_integer | unsigned_integer_64 | vector_2d | vector_3d}* And Dimension *Dimension* [On *Part_list. . .*]

Summary

Creates an Element Field name field_name on the region.

Parameter	Value	Default
Field_name	string	–
Option	{asym_tensor_03 | complex | full_tensor_22 | full_tensor_36 | integer | long_integer | matrix_22 | matrix_33 | real | sym_tensor_21 | sym_tensor_31 | sym_tensor_33 | unsigned_integer | unsigned_integer_64 | vector_2d | vector_3d}	–
Dimension	integer	–

Create Face Field

Syntax

Create Face Field *Field_name* Of Type *{asym_tensor_03 | complex | full_tensor_22 | full_tensor_36 | integer | long_integer | matrix_22 | matrix_33 | real | sym_tensor_21 | sym_tensor_31 | sym_tensor_33 | unsigned_integer | unsigned_integer_64 | vector_2d | vector_3d}* And Dimension *Dimension* [On *Part_list. . .*]

Summary

Creates a face Field named field_name on the region.

Parameter	Value	Default
Field_name	string	–
Option	{asym_tensor_03 | complex | full_tensor_22 | full_tensor_36 | integer | long_integer | matrix_22 | matrix_33 | real | sym_tensor_21 | sym_tensor_31 | sym_tensor_33 | unsigned_integer | unsigned_integer_64 | vector_2d | vector_3d}	–
Dimension	integer	–

Create Nodal Field

Syntax

Create Nodal Field *Field_name* Of Type *{asym_tensor_03 | complex | full_tensor_22 | full_tensor_36 | integer | long_integer | matrix_22 | matrix_33 | real | sym_tensor_21 | sym_tensor_31 | sym_tensor_33 | unsigned_integer | unsigned_integer_64 | vector_2d | vector_3d}* And Dimension *Dimension* [On *Part_list. . .*]

Summary

Creates a Nodal Field name field_name on the region.

Parameter	Value	Default
Field_name	string	–
Option	{asym_tensor_03 | complex | full_tensor_22 | full_tensor_36 | integer | long_integer | matrix_22 | matrix_33 | real | sym_tensor_21 | sym_tensor_31 | sym_tensor_33 | unsigned_integer | unsigned_integer_64 | vector_2d | vector_3d}	–
Dimension	integer	–

Display Truncation Error For

Syntax

Display Truncation Error For *Solution_dof* [With *HowMany* Entries]

Summary

Provides ability to request log file output of the maximum computed truncation error values.

Description

This command allows one to request the maximum magnitudes of predictor error at a small number of DOF to the log file. For vector Fields only the magnitude over all components is considered. Note that additional log file output can be IO intensive so use of this command line should be restricted to debugging of a model.

Default is 5 values.

Parameter	Value	Default
Solution_dof	string	–

Enable Rebalance

Syntax

Enable Rebalance [With Threshold *{=}* *Threshold*]

Summary

Activate rebalance, optionally specifying the rebalance threshold.

Threshold is in terms of the ratio of the maximum (over all processors) cpu time per timestep divided by the average cpu time per timestep. If the observed value of this ratio meets or exceeds the given value, rebalance will be performed.

The default threshold value is 1.25.

Eq

Syntax

Eq *EquationName* For *AssociatedDoF* *{on}* *MeshPartIdentifier* Using *ElementType* With *Terms. . .*

Summary

Specifies an equation to solve. User provides the degree of freedom associated with the equation, the element type, the mesh part (element block) on which to apply the equation and the terms in the equation that are active.

Parameter	Value	Default
EquationName	string	–
AssociatedDoF	string	–
{on}	{@ | at | for | in | on | over}	–
MeshPartIdentifier	string	–
ElementType	string	–
Terms	string. . .	–

Eq

Syntax

Eq Ampere For *Associated DoF* ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* Using *Element Type* With *Terms. . .*

Parameter	Value	Default
Associated DoF	string	–
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
Element Type	string	–
Terms	{adv | diff | lumped_mass | mass | src | xfer}	–

Eq

Syntax

Eq Artificial Compressibility For *Associated DoF* ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* Using *Element Type* With *Terms. . .*

Parameter	Value	Default
Associated DoF	string	–
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
Element Type	string	–
Terms	{diff | div | lumped_mass | mass | xfer}	–

Eq

Syntax

Eq Brinkman Momentum For *Associated DoF* ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* Using *Element Type* With *Terms. . .*

Parameter	Value	Default
Associated DoF	string	–
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
Element Type	string	–
Terms	{adv | diff | lumped_mass | mass | psrc | src | supg | xfer}	–

Eq

Syntax

Eq Charge Density For *Associated DoF* ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* Using *Element Type* With *Terms. . .*

Parameter	Value	Default
Associated DoF	string	–
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
Element Type	string	–
Terms	{adv | diff | lumped_mass | mass | src | supg | xfer}	–

Eq

Syntax

Eq Continuity For *Associated DoF* ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* Using *Element Type* With *Terms. . .*

Parameter	Value	Default
Associated DoF	string	–
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
Element Type	string	–
Terms	{adv | diff | disp_div_hack | div | grad_jump_penalty | lumped_mass | mass | src | stab_darcy_hvm | stab_linear_elasticity_hvm | stab_pgps | stab_ppps | stab_pspg | stab_pspp | xfer}	–

Eq

Syntax

Eq Current For *Associated DoF* ([*{of} SpeciesName*] | [*{in}*

MaterialPhaseName] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* Using *Element Type* With *Terms. . .*

Parameter	Value	Default
Associated DoF	string	–
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
Element Type	string	–
Terms	{diff | lumped_mass | mass | src | xfer}	–

Eq

Syntax

Eq Cvfem Continuity For *Associated DoF* ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* Using *Element Type* With *Terms. . .*

Parameter	Value	Default
Associated DoF	string	–
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
Element Type	string	–
Terms	{adv | ev_adv | lumped_mass | mass | src | xfer}	–

Eq

Syntax

Eq Cvfem Dispersed Continuity For *Associated DoF* ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* Using *Element Type* With *Terms. . .*

Parameter	Value	Default
Associated DoF	string	–
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
Element Type	string	–
Terms	{adv | lumped_mass | mass | src | xfer}	–

Eq

Syntax

Eq Cvfem Dispersed Momentum For *Associated DoF* ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* Using *Element Type* With *Terms. . .*

Parameter	Value	Default
Associated DoF	string	–
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
Element Type	string	–
Terms	{adv | diff | drag | lumped_mass | mass | src | sucv | xfer}	–

Eq

Syntax

Eq Cvfem Div Projection For *Associated DoF* ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* Using *Element Type* With *Terms. . .*

Parameter	Value	Default
Associated DoF	string	–
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
Element Type	string	–
Terms	{def | diff | xfer}	–

Eq

Syntax

Eq Cvfem Energy For *Associated DoF* ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* Using *Element Type* With *Terms. . .*

Parameter	Value	De-fault
Associated DoF	string	–
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhase-Name	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
Element Type	string	–
Terms	{adv | diff | grad_jump_penalty | lumped_mass | mass | src | xfer}	–

Eq

Syntax

Eq Cvfem Lagged Muscl Projection For *Associated DoF* ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* Using *Element Type* With *Terms. . .*

Parameter	Value	Default
Associated DoF	string	–
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
Element Type	string	–
Terms	{def | xfer}	–

Eq

Syntax

Eq Cvfem Lagged Projection For *Associated DoF* ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* Using *Element Type* With *Terms. . .*

Parameter	Value	Default
Associated DoF	string	–
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
Element Type	string	–
Terms	{def | scs_proj | xfer}	–

Eq

Syntax

Eq Cvfem Level Set For *Associated DoF* ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* Using *Element Type* With *Terms. . .*

Parameter	Value	Default
Associated DoF	string	–
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
Element Type	string	–
Terms	{adv | lumped_mass | mass | nc_adv | src | sucv | xfer}	–

Eq

Syntax

Eq Cvfem Lumped Div Projection For *Associated DoF* ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* Using *Element Type* With *Terms. . .*

Parameter	Value	Default
Associated DoF	string	–
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
Element Type	string	–
Terms	{def | diff | xfer}	–

Eq

Syntax

Eq Cvfem Lumped Muscl Projection For *Associated DoF* ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* Using *Element Type* With *Terms. . .*

Parameter	Value	Default
Associated DoF	string	–
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
Element Type	string	–
Terms	{def | xfer}	–

Eq

Syntax

Eq Cvfem Lumped Projection For *Associated DoF* ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* Using *Element Type* With *Terms. . .*

Parameter	Value	Default
Associated DoF	string	–
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
Element Type	string	–
Terms	{def | scs_proj | xfer}	–

Eq

Syntax

Eq Cvfem Mass Fraction For *Associated DoF* ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* Using *Element Type* With *Terms. . .*

Parameter	Value	Default
Associated DoF	string	–
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
Element Type	string	–
Terms	{adv | diff | fracbal | lumped_mass | mass | src | xfer}	–

Eq

Syntax

Eq Cvfem Mesh For *Associated DoF* ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* Using *Element Type* With *Terms. . .*

Parameter	Value	Default
Associated DoF	string	–
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
Element Type	string	–
Terms	{diff | ref_diff | src | xfer}	–

Eq

Syntax

Eq Cvfem Mixture Fraction For *Associated DoF* ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* Using *Element Type* With *Terms. . .*

Parameter	Value	Default
Associated DoF	string	–
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
Element Type	string	–
Terms	{adv | adv_diff | diff | fracbal | lumped_mass | mass | src | sucv | xfer}	–

Eq

Syntax

Eq Cvfem Momentum For *Associated DoF* ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* Using *Element Type* With *Terms. . .*

Parameter	Value	Default
Associated DoF	string	–
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
Element Type	string	–
Terms	{adv | adv_diff | diff | ev_adv_diff | lumped_mass | mass | src | sucv | xfer}	–

Eq

Syntax

Eq Cvfem Muscl Projection For *Associated DoF* ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* Using *Element Type* With *Terms. . .*

Parameter	Value	Default
Associated DoF	string	–
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
Element Type	string	–
Terms	{def | xfer}	–

Eq

Syntax

Eq Cvfem Projection For *Associated DoF* ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* Using *Element Type* With *Terms. . .*

Parameter	Value	Default
Associated DoF	string	–
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
Element Type	string	–
Terms	{def | scs_proj | xfer}	–

Eq

Syntax

Eq Cvfem Phase Balance For *Associated DoF* ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* Using *Element Type* With *Terms. . .*

Parameter	Value	Default
Associated DoF	string	–
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
Element Type	string	–
Terms	{fracbal | src | xfer}	–

Eq

Syntax

Eq Cvfem Quasi Implicit Muscl Projection For *Associated DoF* ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* Using *Element Type* With *Terms...*

Parameter	Value	Default
Associated DoF	string	–
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
Element Type	string	–
Terms	{def | xfer}	–

Eq

Syntax

Eq Cvfem Solvent Continuity For *Associated DoF* ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* Using *Element Type* With *Terms...*

Parameter	Value	Default
Associated DoF	string	–
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
Element Type	string	–
Terms	{adv | lumped_mass | mass | src | xfer}	–

Eq

Syntax

Eq Cvfem Solvent Momentum For *Associated DoF* ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* Using *Element Type* With *Terms. . .*

Parameter	Value	Default
Associated DoF	string	–
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
Element Type	string	–
Terms	{adv | diff | drag | lumped_mass | mass | src | sucv | xfer}	–

Eq

Syntax

Eq Cvfem Specific Dissipation Rate For *Associated DoF* ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* Using *Element Type* With *Terms. . .*

Parameter	Value	Default
Associated DoF	string	–
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
Element Type	string	–
Terms	{adv | diff | lumped_mass | mass | src | xfer}	–

Eq

Syntax

Eq Cvfem Turbulent Kinetic Energy For *Associated DoF* ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* Using *Element Type* With *Terms. . .*

Parameter	Value	Default
Associated DoF	string	–
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
Element Type	string	–
Terms	{adv | diff | lumped_mass | mass | src | xfer}	–

Eq

Syntax

Eq Cvfem Turbulence Dissipation Rate For *Associated DoF* ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* Using *Element Type* With *Terms. . .*

Parameter	Value	Default
Associated DoF	string	–
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
Element Type	string	–
Terms	{adv | diff | lumped_mass | mass | src | xfer}	–

Eq

Syntax

Eq Cvfem Velocity Pressure Projection For *Associated DoF* ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* Using *Element Type* With *Terms. . .*

Parameter	Value	Default
Associated DoF	string	–
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
Element Type	string	–
Terms	{lumped_mass | mass | src | xfer}	–

Eq

Syntax

Eq Darcy Momentum For *Associated DoF* ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* Using *Element Type* With *Terms. . .*

Parameter	Value	Default
Associated DoF	string	–
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
Element Type	string	–
Terms	{adv | diff | src | stab_darcy_hvm | xfer}	–

Eq

Syntax

Eq Div Projection For *Associated DoF* ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* Using *Element Type* With *Terms. . .*

Parameter	Value	Default
Associated DoF	string	–
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
Element Type	string	–
Terms	{def | xfer}	–

Eq

Syntax

Eq Energy For *Associated DoF* ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* Using *Element Type* With *Terms. . .*

Parameter	Value	Default
Associated DoF	string	–
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
Element Type	string	–
Terms	{adv | diff | div | ggls | grad_jump_penalty | lumped_mass | mass | src | supg | xfer}	–

Eq

Syntax

Eq Enthalpy For *Associated DoF* ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* Using *Element Type* With *Terms. . .*

Parameter	Value	Default
Associated DoF	string	–
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
Element Type	string	–
Terms	{adv | diff | div | fracbal | grad_jump_penalty | lumped_mass | mass | psrc | src | stab_darcy_hvm | sum | supg | upwind_adv | xfer}	–

Eq

Syntax

Eq Extension Speed For *Associated DoF* ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* Using *Element Type* With *Terms. . .*

Parameter	Value	Default
Associated DoF	string	–
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
Element Type	string	–
Terms	{def | src | xfer}	–

Eq

Syntax

Eq Hdiff For *Associated DoF* ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* Using *Element Type* With *Terms. . .*

Parameter	Value	Default
Associated DoF	string	–
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
Element Type	string	–
Terms	{diff | lumped_mass | mass | src | supg | xfer}	–

Eq

Syntax

Eq Lagged Projection For *Associated DoF* ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* Using *Element Type* With *Terms. . .*

Parameter	Value	Default
Associated DoF	string	–
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
Element Type	string	–
Terms	{def | xfer}	–

Eq

Syntax

Eq Level Set For *Associated DoF* ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* Using *Element Type* With *Terms. . .*

Parameter	Value	De-fault
Associated DoF	string	–
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhase-Name	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
Element Type	string	–
Terms	{adv | diff | lumped_mass | mass | src | supg | taylor_galerkin | xfer}	–

Eq

Syntax

Eq Lubrication For *Associated DoF* ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* Using *Element Type* With *Terms. . .*

Parameter	Value	Default
Associated DoF	string	–
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
Element Type	string	–
Terms	{bound | def | diff | mass | src | xfer}	–

Eq

Syntax

Eq Lubrication Height For *Associated DoF* ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* Using *Element Type* With *Terms. . .*

Parameter	Value	Default
Associated DoF	string	–
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
Element Type	string	–
Terms	{def | xfer}	–

Eq

Syntax

Eq Lumped Div Projection For *Associated DoF* ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* Using *Element Type* With *Terms. . .*

Parameter	Value	Default
Associated DoF	string	–
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
Element Type	string	–
Terms	{def | xfer}	–

Eq

Syntax

Eq Lumped Projection For *Associated DoF* ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* Using *Element Type* With *Terms. . .*

Parameter	Value	Default
Associated DoF	string	–
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
Element Type	string	–
Terms	{def | xfer}	–

Eq

Syntax

Eq Magnetic Continuity For *Associated DoF* ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* Using *Element Type* With *Terms. . .*

Parameter	Value	Default
Associated DoF	string	–
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
Element Type	string	–
Terms	{div | src | stab_mag | xfer}	–

Eq

Syntax

Eq Magnetic Flux For *Associated DoF* ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* Using *Element Type* With *Terms. . .*

Parameter	Value	Default
Associated DoF	string	–
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
Element Type	string	–
Terms	{adv | diff | lumped_mass | mass | src | xfer}	–

Eq

Syntax

Eq Magnetic Flux Intensity For *Associated DoF* ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* Using *Element Type* With *Terms. . .*

Parameter	Value	Default
Associated DoF	string	–
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
Element Type	string	–
Terms	{diff}	–

Eq

Syntax

Eq Mass Balance For *Associated DoF* ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* Using *Element Type* With *Terms. . .*

Parameter	Value	Default
Associated DoF	string	–
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
Element Type	string	–
Terms	{adv | diff | div | fracbal | grad_jump_penalty | lumped_mass | mass | psrc | src | stab_darcy_hvm | sum | supg | upwind_adv | xfer}	–

Eq

Syntax

Eq Mesh For *Associated DoF* ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* Using *Element Type* With *Terms. . .*

Parameter	Value	Default
Associated DoF	string	–
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
Element Type	string	–
Terms	{def | diff | ref_diff | src | xfer}	–

Eq

Syntax

Eq Mixture Fraction For *Associated DoF* ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* Using *Element Type* With *Terms. . .*

Parameter	Value	Default
Associated DoF	string	–
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
Element Type	string	–
Terms	{adv | diff | mass | src | supg | xfer}	–

Eq

Syntax

Eq Momentum For *Associated DoF* ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* Using *Element Type* With *Terms. . .*

Parameter	Value	De-fault
Associated DoF	string	–
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhase-Name	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
Element Type	string	–
Terms	{adv | diff | drag | grad_jump_penalty | lumped_mass | mass | psrc | src | supg | xfer}	–

Eq

Syntax

Eq Bar Momentum For *Associated DoF* ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* Using *Element Type* With *Terms. . .*

Parameter	Value	Default
Associated DoF	string	–
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
Element Type	string	–
Terms	{adv | src | xfer}	–

Eq

Syntax

Eq Neutron For *Associated DoF* ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* Using *Element Type* With *Terms. . .*

Parameter	Value	Default
Associated DoF	string	–
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
Element Type	string	–
Terms	{diff | mass | src | xfer}	–

Eq

Syntax

Eq Porous Enthalpy For *Associated DoF* ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* Using *Element Type* With *Terms. . .*

Parameter	Value	Default
Associated DoF	string	–
{of}	{of | species | subindex}	–
Species-Name	string	–
{in}	{in | material_phase}	–
MaterialPhase-Name	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
Element Type	string	–
Terms	{adv | diff | div | fracbal | grad_jump_penalty | lumped_mass | mass | psrc | src | stab_darcy_hvm | sum | supg | upwind_adv | xfer}	–

Eq

Syntax

Eq Porous Species For *Associated DoF* ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* Using *Element Type* With *Terms. . .*

Parameter	Value	Default
Associated DoF	string	–
{of}	{of | species | subindex}	–
Species-Name	string	–
{in}	{in | material_phase}	–
MaterialPhase-Name	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
Element Type	string	–
Terms	{adv | diff | div | fracbal | grad_jump_penalty | lumped_mass | mass | psrc | src | stab_darcy_hvm | sum | supg | upwind_adv | xfer}	–

Eq

Syntax

Eq Potential For *Associated DoF* ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* Using *Element Type* With *Terms. . .*

Parameter	Value	Default
Associated DoF	string	–
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
Element Type	string	–
Terms	{def | src | xfer}	–

Eq

Syntax

Eq Precursor Conservation For *Associated DoF* ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* Using *Element Type* With *Terms. . .*

Parameter	Value	Default
Associated DoF	string	–
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
Element Type	string	–
Terms	{mass | src | xfer}	–

Eq

Syntax

Eq Projection For *Associated DoF* ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* Using *Element Type* With *Terms. . .*

Parameter	Value	Default
Associated DoF	string	–
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
Element Type	string	–
Terms	{def | xfer}	–

Eq

Syntax

Eq Quasi Implicit Projection For *Associated DoF* ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* Using *Element Type* With *Terms. . .*

Parameter	Value	Default
Associated DoF	string	–
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
Element Type	string	–
Terms	{def | xfer}	–

Eq

Syntax

Eq Radiation Transport For *Associated DoF* ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* Using *Element Type* With *Terms. . .*

Parameter	Value	Default
Associated DoF	string	–
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
Element Type	string	–
Terms	{xfer}	–

Eq

Syntax

Eq Semilagrangian Level Set For *Associated DoF* ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* Using *Element Type* With *Terms. . .*

Parameter	Value	Default
Associated DoF	string	–
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
Element Type	string	–
Terms	{def | xfer}	–

Eq

Syntax

Eq Solid For *Associated DoF* ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* Using *Element Type* With *Terms. . .*

Parameter	Value	Default
Associated DoF	string	–
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
Element Type	string	–
Terms	{adv | diff | drag | mass | ref_diff | src | xfer}	–

Eq

Syntax

Eq Sp For *Associated DoF* ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* Using *Element Type* With *Terms. . .*

Parameter	Value	Default
Associated DoF	string	–
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
Element Type	string	–
Terms	{adv | diff | src | xfer}	–

Eq

Syntax

Eq Species For *Associated DoF* ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* Using *Element Type* With *Terms. . .*

Parameter	Value	De-fault
Associated DoF	string	–
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
Material-PhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
Element Type	string	–
Terms	{adv | diff | div | fracbal | grad_jump_penalty | lumped_mass | mass | src | supg | taylor_galerkin | xfer}	–

Eq

Syntax

Eq Suspension For *Associated DoF* ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* Using *Element Type* With *Terms. . .*

Parameter	Value	Default
Associated DoF	string	–
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
Element Type	string	–
Terms	{adv | diff | lumped_mass | mass | src | supg | xfer}	–

Eq

Syntax

Eq User Scalar For *Associated DoF* ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* Using *Element Type* With *Terms. . .*

Parameter	Value	Default
Associated DoF	string	–
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
Element Type	string	–
Terms	{xfer}	–

Eq

Syntax

Eq User Tensor For *Associated DoF* ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* Using *Element Type* With *Terms. . .*

Parameter	Value	Default
Associated DoF	string	–
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
Element Type	string	–
Terms	{xfer}	–

Eq

Syntax

Eq User Vector For *Associated DoF* ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* Using *Element Type* With *Terms. . .*

Parameter	Value	Default
Associated DoF	string	–
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
Element Type	string	–
Terms	{xfer}	–

Eq

Syntax

Eq Voltage For *Associated DoF* ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* Using *Element Type* With *Terms. . .*

Parameter	Value	Default
Associated DoF	string	–
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
Element Type	string	–
Terms	{diff | psrc | src | xfer}	–

Eq Scaled_Sp

Syntax

Eq Scaled_Sp *Order* For *AssociatedDoF {on} MeshPartIdentifier* Using *ElementType* [With Src]

Summary

Specifies the scaled Spherical Harmonics equation set to solve for Participating Media Radiation (PMR). User provides the order, degree of freedom associated with the equation, the element type and the mesh part (element block) on which to apply the equation.

Parameter	Value	Default
Order	integer	–
AssociatedDoF	string	–
{on}	{@ | at | for | in | on | over}	–
MeshPartIdentifier	string	–
ElementType	string	–

Global Variables Excluded From Log File

Syntax

Global Variables Excluded From Log File *{=} variable_names. . .*

Summary

Specify global variables that should not have their current values output to the log file at the end of every time step.

Description

By default most global variables will be output. The list of arguments are treated as regular expressions using the grep syntax. For example, “GLOBAL VARIABLES EXCLUDED FROM LOG FILE = .*” will cause no global variables to be output to the log file, and “LOG FILE GLOBAL VARIABLES = encl.*” will exclude all global variables starting with “encl” from the log file

output.

Parameter	Value	Default
{=}	{= | are | is}	–
variable_names	string. . .	–

Integration Rule For

Syntax

Integration Rule For *BlockName {=} Order*

Summary

Attach an integration rule to a mesh part

Parameter	Value	Default
BlockName	string	–
{=}	{= | are | is}	–
Order	integer	–

List Mesh Groups

Syntax

List Mesh Groups [*Name. . .*]

Summary

This command is deprecated and no longer has any effect. Mesh group summaries are always printed to the log file.

Maximum Number Of Nodal Rebalance Iterations

Syntax

Maximum Number Of Nodal Rebalance Iterations *{=} Count*

Summary

Specify the maximum number of nodal rebalance iterations to be performed after a dynamic rebalance.

Parameter	Value	Default
{=}	{= | are | is}	–
Count	integer	–

Maximum Number Of Rebalances

Syntax

Maximum Number Of Rebalances *{=} Count*

Summary

Specify the maximum number of times to perform a mesh rebalance.

Parameter	Value	Default
{=}	{= | are | is}	–
Count	integer	–

Mesh Group

Syntax

Mesh Group *name {=} Mesh_extent. . .*

Summary

Specifies a user selected grouping for mesh entities of the same type to be grouped under a single name alias. While defining a simulation model the alias can be used when referring to the entire collection of mesh entities. By default the group name ALL_BLOCKS is always supplied.

Parameter	Value	Default
name	string	–
{=}	{= | are | is}	–
Mesh_extent	string. . .	all_blocks

Minimum Nonlinear Iterations

Syntax

Minimum Nonlinear Iterations *{=} Np*

Summary

Minimum number of actual nonlinear solves that are required.

Description

Each nonlinear iteration involves first assembling the nonlinear residual and Jacobian matrix and second solving the linear system for a nonlinear correction that is applied to the estimate of the nonlinear solution.

The nonlinear solver will continue iterating until this minimum number of solves are performed. The default value is one (1) but setting this to zero can be useful in some situations.

This command is identical to “minimum nonlinear solves = X”.

Parameter	Value	Default
{=}	{= | are | is}	–
Np	integer	1

No Viewfactor Surface Randomize

Syntax

No Viewfactor Surface Randomize

Summary

Turns off viewfactor surface randomization.

Description

Parallel performance of view factor calculation benefits from internally randomizing the location of surface data over the processor decomposition. Use of this command turns off randomization and may result in a reduction in parallel performance.

Nonlinear Relative Correction Tolerance

Syntax

Nonlinear Relative Correction Tolerance *{=}* *Pt*

Summary

Convergence tolerance of nonlinear relative correction norm (relative correction = nonlinear correction norm / nonlinear solution norm) for the iteration. Satisfaction of this criterion is sufficient for the iteration to be considered finished. This criterion is most useful for problems containing DOFs with very different orders of magnitude.

Parameter	Value	Default
{=}	{= | are | is}	–
Pt	real	–

Nonlinear Solver

Syntax

Nonlinear Solver *{=}* *Solver*

Summary

Specifies the nonlinear solver.

Parameter	Value	Default
{=}	{= | are | is}	–
Solver	string	–

Only Compute Viewfactors

Syntax

Only Compute Viewfactors

Summary

Perform viewfactor solves and then terminate the simulation.

Description

Database names for view factor files must be specified in the enclosure definition so they can be written and read in for subsequent analyses.

Phase Change Relaxation Factor

Syntax

Phase Change Relaxation Factor *{=}* *Prf*

Summary

In versions > 4.52 this command has no effect. It was previously used with the deprecated "USE_PHASE_CHANGE" model for specific heat and is now only left for backwards compatibility.

Parameter	Value	Default
{=}	{= | are | is}	–
Prf	real	0.5

Point Source For

Syntax

Point Source For *EquationName* ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*]) *{on}* *MeshPartIdentifier* Value *{=}* *PointSourceValue* X *{=}* *Xcoord* [Y *{=}* *Ycoord* Z *{=}* *Zcoord*]

Summary

Adds a point source of value PointSourceValue at position (X,Y,Z) for the identified equation. Currently only works for scalars.

Parameter	Value	Default
EquationName	string	–
{of}	{of | species}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{on}	{@ | at | for | in | on | over}	–
MeshPartIdentifier	string	–
{=}	{= | are | is}	–
PointSourceValue	real	–
{=}	{= | are | is}	–
Xcoord	real	–

Post Process

Syntax

Post Process *{integral | normalized}* *VariableName* On *IoPartList...* As *OutputName* [Using *WeightList...* Weight]

Summary

Post process an Aria variable and save it to a global variable.

Description

Post process Aria variable VariableName and save it to a global variable OutputName. The global OutputName will be written to the log file and is also available for output to an ExodusII database. During post processing the variable a product is formed from VariableName and the evaluation of WeightList items. The keyword WEIGHT terminates the Weightlist items.

Parameter	Value	Default
PostProcessType	{integral | normalized}	–
VariableName	string	–
IoPartList	string. . .	–
OutputName	string	–

Post Process Flux

Syntax

Post Process Flux *variableName* On *ioPartList. . .* As *outputName* [Restricted To *restrictPartList. . .*]

Summary

Post process an Aria flux and save the integrated results to a global Aria Field.

Description

Post process Aria flux FluxName and save the integrated flux to a global Field variable OutputName. The resultant variable OutputName will be available for output to an ExodusII database. Examples of FluxName are heat_conduction and current_density.

Parameter	Value	Default
variableName	string	–
ioPartList	string. . .	–
outputName	string	–

Post Process Nodal

Syntax

Post Process Nodal *{integral | normalized} VariableName* On *IoPartList. . .* As *OutputName* [Using *WeightList. . .* Weight]

Summary

Post process an Aria nodal variable and save the results to another Aria Field.

Description

Post process Aria variable VariableName and save it to another nodal Field variable OutputName. The resultant variable OutputName will be available for output to an ExodusII database. PostProcessType is NORMALIZED so that during post processing the variable a product is formed from VariableName and

the evaluation of WeightList items. The keyword WEIGHT terminates the Weightlist items.

Parameter	Value	Default
PostProcessType	{integral | normalized}	–
VariableName	string	–
IoPartList	string. . .	–
OutputName	string	–

Predictor Fields

Syntax

Predictor Fields *{=} Field_names. . .*

Summary

Specification of active fields to be used in predictor algorithm.

Description

Specifies which fields to examine or ignore in the algorithm for adaptive time step selection. Fields that are *not* predictor fields will not be predicted solutions for the first two time steps. This is important for fields like pressure and electrostatic potential which may exhibit large jumps in their solutions between the initial conditions and the first solutions owing to the absence of time dependent terms in their governing equations during startup.

The selected fields will contribute to the d_{n+1} norm discussed in the `TIME TRUNCATION ERROR` time block command. This command line can be provided multiple times with cumulative results.

By default all fields are included. All field names following the optional "NOT" keyword will be excluded from the selection algorithm.

Parameter	Value	Default
{=}	{= | are | is}	–
Field_names	string. . .	–

Preprocess Enclosure Topology

Syntax

Preprocess Enclosure Topology [*Options. . .*]

Summary

Process enclosures to the extent that their topology is defined. If a topology file is requested within the enclosure definition the file will be output.

Preprocess Enclosures

Syntax

Preprocess Enclosures [*Options. . .*]

Summary

Preprocess enclosures subject to options: ORIGINAL_FACES, ENCLOSURE_FACETS, ENCLOSURE_FACET_EDGES, CONTACT_FACETS, CONTACT_FACET_EDGES and COARSE_MESH. When invoked at Region preprocessing is performed and the simulation will terminate.

Description

For debug and inspection purposes the surfaces defined in the enclosure definitions will be processed, then output to the exodus database before terminating the simulation. If no options are chosen the default is to select all the available information. Options are: ORIGINAL_FACES, ENCLOSURE_FACETS, ENCLOSURE_FACET_EDGES, CONTACT_FACETS, CONTACT_FACET_EDGES and COARSE_MESH. Facets are output as triangle elements, edges as bars and faces as shell elements.

Pressure Stabilization

Syntax

Pressure Stabilization *Is {no_stabilization | pgps_dt | ppps_constant | pspg_constant | pspg_global | pspg_local | pspg_projected_shakib | pspg_projected_unsteady_shakib | pspg_shakib | pspg_unsteady | pspg_unsteady_shakib | pspp_constant | pspp_off_interface | pspp_transient} With Scaling {=} ScalingParam*

Summary

Specifies the pressure stabilization type and scaling value.

Parameter	Value	Default
Is	string	–
PStabMethod	{no_stabilization | pgps_dt | ppps_constant | pspg_constant | pspg_global | pspg_local | pspg_projected_shakib | pspg_projected_unsteady_shakib | pspg_shakib | pspg_unsteady | pspg_unsteady_shakib | pspp_constant | pspp_off_interface | pspp_transient}	–
With	string	–
Scaling	string	–
{=}	{= | are | is}	–
ScalingParam	real	–

Rebalance After Adaptivity

Syntax

Rebalance After Adaptivity

Summary

Command to automatically execute a rebalance after each solution control adaptivity step

Rebalance Load Measure Type

Syntax

Rebalance Load Measure Type *{=}* *MeasureType*

Summary

Specify the measure used to evaluate the load for rebalancing.

Parameter	Value	Default
{=}	{= | are | is}	–
MeasureType	string	–

Rebalance Time Step Frequency

Syntax

Rebalance Time Step Frequency *{=}* *Frequency*

Summary

Time step frequency for running rebalance.

Parameter	Value	Default
{=}	{= | are | is}	–
Frequency	integer	–

Residual Weight

Syntax

Residual Weight For *dof_name* *{=}* *weight*

Summary

Specifies how to scale the nonlinear residuals.

Description

This option will only affect problems using Tpetra-based linear solvers and CUSTOM nonlinear residual scaling

Parameter	Value	Default
dof_name	string	–
{=}	{= | are | is}	–
weight	real	–

Root Finder Attempt Flux Solve Before Bisection

Syntax

Root Finder Attempt Flux Solve Before Bisection *{=}* *{false | no | off | on | true | yes}*

Summary

Specify whether or not to attempt a solve using a flux BC before trying the bisection. Solving using a flux BC is often much faster so this will only run the slower bisection method when that results in a failed time step.

Parameter	Value	Default
{=}	{= | are | is}	–
Bool	{false | no | off | on | true | yes}	–

Root Finder Controls Bc For Equation

Syntax

Root Finder Controls Bc For Equation *EquationName* *{on}* *MeshPartIdentifier* Initialvalue *{=}* *InitialValue*

Summary

Specify which Dirichlet BC the root finding algorithm should control and the initial guess for the applied value.

Parameter	Value	Default
EquationName	string	–
{on}	{@ | at | for | in | on | over}	–
MeshPartIdentifier	string	–
{=}	{= | are | is}	–
InitialValue	real	–

Root Finder Correlation Between Flux And Dirichlet Is Positive

Syntax

Root Finder Correlation Between Flux And Dirichlet Is Positive *{=}* *{false | no | off | on | true | yes}*

Summary

Specify whether an increase in the applied Dirichlet value will cause an increase in the resulting flux.

Parameter	Value	Default
{=}	{= | are | is}	–
CorrelationBool	{false | no | off | on | true | yes}	FALSE

Root Finder Dirichlet Tolerance Is

Syntax

Root Finder Dirichlet Tolerance Is *DirichletTolerance*

Summary

Specify the absolute tolerance on the applied Dirichlet value that the root finder will attempt to converge to. The root finder will end when either the flux or Dirichlet tolerance is reached.

Parameter	Value	Default
DirichletTolerance	real	–

Root Finder Flux Tolerance Is

Syntax

Root Finder Flux Tolerance Is *FluxTolerance*

Summary

Specify the absolute flux tolerance that the root finder will attempt to converge to. The root finder will end when either the flux or Dirichlet tolerance is reached.

Parameter	Value	Default
FluxTolerance	real	–

Root Finder Target Flux Expressions Are

Syntax

Root Finder Target Flux Expressions Are *FluxExpressionNames. . .*

Summary

Specify contributions to the target flux for the root finder boundary. Contributions are summed and integrated over the surface.

Parameter	Value	Default
FluxExpressionNames	string. . .	–

Root Finder Target Flux Value Is

Syntax

Root Finder Target Flux Value Is *FluxTarget*

Summary

Specify the target value for the area-averaged flux contributions from the specified expressions.

Parameter	Value	Default
FluxTarget	real	–

Save Coeff

Syntax

Save Coeff *{all_numerical_methods | ccfvm | cvfem | galerkin_fem}* Expression *ExpressionName* ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*]) As *FieldName* [On Parts *partList. . .*]

Summary

Save off a particular expression for a particular formulation

Parameter	Value	Default
NumericalMethod	{all_numerical_methods | ccfvm | cvfem | galerkin_fem}	–
ExpressionName	string	–
{of}	{of | species}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
FieldName	string	–

Save Ip

Syntax

Save Ip *{ccfvm | cvfem | galerkin_fem | scs_cvfem | scv_cvfem}* Expression *ExpressionName* ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*]) As *FieldName* On Parts *partList. . .*

Summary

Save off a particular integration point expression for a particular formulation

Description

Saves off an expression at integration points on one or more mesh parts. The expression can later be re-used using the “USE IP” command. The list of mesh parts should include all blocks and surfaces on which the expression will be needed, but can also include “all” to save on all blocks and surfaces.

Parameter	Value	Default
NumericalIp	{ccfvm | cvfem | galerkin_fem | scs_cvfem | scv_cvfem}	–
ExpressionName	string	–
{of}	{of | species}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
FieldName	string	–
partList	string. . .	–

Save Residuals

Syntax

Save Residuals *{=}* *mode*

Summary

Causes Aria to save the residuals to a field with the prefix residual->DOF-NAME. This will be done for all fields (though we could make it a per-field option). Valid choices are OFF (default), BEFORE_BCS and AFTER_BCS.

Parameter	Value	Default
{=}	{= | are | is}	–
mode	string	OFF

Segregated Global Minimum Convergence Rate

Syntax

Segregated Global Minimum Convergence Rate *{=}* *MinRate* [Number Of Steps *{=}* *MinSteps*]

Summary

Set a minimum convergence rate based on the initial nonlinear residual of each equation system in the problem. At each segregated iteration the equation system with the largest initial residual is tested using

$$r_N / r_{N-\text{MinSteps}} < \text{MinRate}^{\text{MinSteps}}$$

If this condition is not met then the time step fails and is retried with a shorter dt.

Parameter	Value	Default
{=}	{= | are | is}	–
MinRate	real	–

Segregated Global Nonlinear Initial Residual Tolerance

Syntax

Segregated Global Nonlinear Initial Residual Tolerance *{=}* *Pt*

Summary

Convergence tolerance for global segregated nonlinear residual for the iteration. The iteration is considered finished when all of the individual equation systems have an initial nonlinear residual less than the given tolerance. This is cheaper than other tests that require recomputing the nonlinear residual. However, care should be taken since the initial nonlinear residual of the individual equation systems might not be a good measure of the final nonlinear residual of those systems due to deltas produced by other equation systems.

Parameter	Value	Default
{=}	{= | are | is}	–
Pt	real	–

Segregated Global Nonlinear Number Of Steps

Syntax

Segregated Global Nonlinear Number Of Steps *{=}* *Number_of_steps* [Fail Step *{=}* *{false \| no \| off \| on \| true \| yes}*]

Summary

Maximum number of segregated solution steps within a time step. Depending on the fail value the solution will either be accepted, or the time step will fail when the maximum number of steps is reached.

Parameter	Value	Default
{=}	{= | are | is}	–
Number_of_steps	integer	–

Segregated Global Nonlinear Relative Residual Tolerance

Syntax

Segregated Global Nonlinear Relative Residual Tolerance *{=}* *Pt*

Summary

Convergence tolerance for global segregated nonlinear residual for the iteration. Satisfaction of this criterion is sufficient for the iteration to be considered finished. This line command can be used when the user desires to enforce a global relative residual tolerance for the entire segregated system that may be different from the individual equation systems value.

Parameter	Value	Default
{=}	{= | are | is}	–
Pt	real	–

Segregated Global Nonlinear Residual Tolerance

Syntax

Segregated Global Nonlinear Residual Tolerance *{=}* *Pt* [Minimum Steps *{=}* *minStepsValue*]

Summary

Convergence tolerance for global segregated nonlinear residual for the iteration. Satisfaction of this criterion is sufficient for the iteration to be considered finished. This line command can be used when the user desires to enforce a global residual enforcement for the entire segregated system that may be different from the individual equation systems value

Parameter	Value	Default
{=}	{= | are | is}	–
Pt	real	–

Segregated Global Nonlinear Target Number Of Steps Per Time Step

Syntax

Segregated Global Nonlinear Target Number Of Steps Per Time Step *{=}* *target_iterations*

Summary

Set a target number of segregated iterations per time step to use for time step control. If more than the specified number of iterations are required to converge a time step the next time step size will be decreased proportional to the ratio of the target number and the number taken. If a maximum number of allowed segregated iterations is set it is recommended to set the target to 1-2 fewer iterations to avoid an excessive number of failed time steps.

Parameter	Value	Default
{=}	{= | are | is}	–
target_iterations	integer	–

Stk Bucket Capacity

Syntax

Stk Bucket Capacity *{=}* *value*

Summary

Developer command for setting a STK fixed bucket capacity.

Parameter	Value	Default
{=}	{= | are | is}	–
value	integer	–

Stk Initial Bucket Capacity

Syntax

Stk Initial Bucket Capacity *{=}* *value*

Summary

Developer command for setting a STK initial bucket capacity. This should be less than the maximum capacity to enable variable-capacity behavior.

Parameter	Value	Default
{=}	{= | are | is}	–
value	integer	–

Stk Maximum Bucket Capacity

Syntax

Stk Maximum Bucket Capacity *{=}* *value*

Summary

Developer command for setting a STK maximum bucket capacity. This should be greater than the minimum capacity to enable variable-capacity behavior.

Parameter	Value	Default
{=}	{= | are | is}	–
value	integer	–

Switch Secondary Solver On Recoverable Linear Solve Failure

Syntax

Switch Secondary Solver On Recoverable Linear Solve Failure

Summary

Switch auxiliary solver on recoverable linear solver failures (max iteration count and loss of accuracy). Only applicable if USE SECONDARY LINEAR SOLVER is present. For additional information, refer to the documentation on the USE SECONDARY LINEAR SOLVER command.

Species

Syntax

Species *Names. . .*

Parameter	Value	Default
Names	string. . .	–

Thermal Strain Reference Temperature

Syntax

Thermal Strain Reference Temperature *{=}* *Value*

Summary

Set the reference temperature for thermal strains.

Parameter	Value	Default
{=}	{= | are | is}	–
Value	real	–

Track Irreversible Phase Change

Syntax

Track Irreversible Phase Change *Name* *Variable* *{< | > | cdfem inequality criterion undefined}* *Value*

Summary

Specify that an interface corresponds to an irreversible phase changed based on the value of a variable. Once the criterion is met an element is moved to the deactivated phase and the field is no longer required to be defined there.

Parameter	Value	Default
Name	string	–
Variable	string	–
Criterion	{< | > | cdfem inequality criterion undefined}	–
Value	real	–

Use Coeff Expression

Syntax

Use Coeff Expression *ExpressionName* ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*]) From *FieldName*

Summary

Override an expression from the given field

Parameter	Value	Default
ExpressionName	string	–
{of}	{of | species}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
FieldName	string	–

Use Dash Enclosures

Syntax

Use Dash Enclosures [*type tol {=} Params. . .*]

Summary

Perform DASH contact search on the enclosure surfaces.

Description

Performing a DASH contact search on the enclosure surfaces will prevent any faces in contact from being processed as part of an enclosure. Optionally one may define a relative tolerance based upon the characteristic element length, or absolute tolerances in positive and negative directions from the enclosure surface.

Use Data Block

Syntax

Use Data Block *Name*

Summary

Reference to predefined data to be used by the user subroutine. These values may be changed by the user subroutine.

Parameter	Value	Default
Name	string	–

Use Ip Expression

Syntax

Use Ip Expression *ExpressionName* ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*]) From *FieldName*

Summary

Override an expression at integration points from the given field

Parameter	Value	Default
ExpressionName	string	–
{of}	{of | species}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
FieldName	string	–

Use Percept For Adaptivity

Syntax

Use Percept For Adaptivity

Summary

Add this command to use percept for mesh adaptivity

User Field

Syntax

User Field *REAL_or_INTEGER NODE_or_FACE_or_ELEMENT SCALAR_or_VECTOR_or_TENSOR Variable_name* On *Mesh_extent* ([Smooth *nsmooth* Iterations] | [Value *{=}* *Magnitude. . .*])

Summary

Defines a user field of standard type data_type (REAL or INTEGER) on the FEM entity (NODE, FACE, or ELEMENT) with the specified name and Sierra field type (SCALAR, VECTOR, or TENSOR). The field will be defined on the specified mesh part or mesh group alias. Normally this field would be supplied to Aria through a transfer. If a number of smoothing iterations is specified, the field will be smoothed this many times after any transfers.

Parameter	Value	Default
REAL_or_INTEGER	string	–
NODE_or_FACE_or_ELEMENT	string	–
SCALAR_or_VECTOR_or_TENSOR	string	–
Variable_name	string	–
Mesh_extent	string	–
nsmooth	integer	0
{=}	{= | are | is}	–
Magnitude	real. . .	–

Use Finite Element Model

Syntax

Use Finite Element Model *ModelName* [Model Coordinates Are *Nodal_variable_name*]

Summary

Associates a predefined finite element model with this region.

Parameter	Value	Default
ModelName	string	–

Vector Point Source For

Syntax

Vector Point Source For *EquationName* ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*]) *{on}* *MeshPartIdentifier* V_X *{=}* *X_PointSourceValue* V_Y *{=}* *Y_PointSourceValue* V_Z *{=}* *Z_PointSourceValue* At X *{=}* *Xcoord* Y *{=}* *Ycoord* Z *{=}* *Zcoord*

Summary

Adds a vector valued point source (v_x, v_y, v_z) at position (X,Y,Z) for the identified equation. Although all three dimensions must be specified, the z values are ignored for 2D problems.

Parameter	Value	Default
EquationName	string	–
{of}	{of | species}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{on}	{@ | at | for | in | on | over}	–
MeshPartIdentifier	string	–
{=}	{= | are | is}	–
X_PointSourceValue	real	–
{=}	{= | are | is}	–
Y_PointSourceValue	real	–
{=}	{= | are | is}	–
Z_PointSourceValue	real	–
{=}	{= | are | is}	–
Xcoord	real	–
{=}	{= | are | is}	–
Ycoord	real	–
{=}	{= | are | is}	–
Zcoord	real	–

Define Global Scalar

Syntax

Define Global Scalar *variable-name. . .* As *{int | real}* Operation *{max | min | sum}* ([Initial Value *initial-value*] | [Reset On *{nonlinear iteration | nonlinear step}*])

Parameter	Value	Default
variable-name	string. . .	–
type	{int | real}	–
op	{max | min | sum}	–
initial-value	real	–
reset-on	{nonlinear iteration | nonlinear step}	–

Initial Uniform Refinement For

Syntax

Initial Uniform Refinement For *Num* Iterations

Summary

Refine for n number of iterations on initialization

Parameter	Value	Default
Num	integer	–

7.19 Advective Bar

Advective Bar Network

Scope

Aria Region, Equation System, Explicit Equation System, Root Finder Equation System

Summary

Defines an advective bar that can be used by a convective flux boundary condition.

Description

Newton's law of cooling specifies that the heat flux normal to a surface is proportional to the difference between the unknown temperature of the surface and some reference temperature of the fluid in which the surface is immersed: $q_n = h(T - T_r)$. The convection coefficient, h, and the reference temperature, T_r, may be specified in several ways. Of particular interest is the situation where the fluid flow can be reduced to a simple 1D flow. In this case, heat transfer between the surface and the fluid is modeled with an advective bar model where each segment on the bar and each facet on the surface assumes a constant heat flux based on a constant h, T and T_r.

```
begin Advective Bar Network Name

   Add Volume Parameters...

   Annulus Diameter Ratio Function {=} Function

   Annulus Diameter Ratio Threshold {=} Threshold

   Coordinate Reference {=} Nodelist Component {s | x | y | z}

   Entrance Effect Starting Node Id {=} NodeID [ Offset {=} Value  ]

   Flow Cross Section Area {=} Value

   Flow Cross Sectional Area Function {=} Function
```

```
   Fluid Density {=} Value

   Fluid Density Function {=} Function

   Hydraulic Diameter Function {=} Function

   Initial Temperature {=} t

   Mass Flow Rate {on} Nodelist {=} Value

   Mass Flow Rate Function {on} Nodelist {=} Function

   Maximum Search Tolerance {=} Tolerance

   Search Interval {=} Interval

   Velocity Bc {on} Nodelist {=} Velocity

   Visualization {off | on}

   Wetted Perimeter Function {=} Function

end Advective Bar Network Name
```

Line Commands

Add Volume

Syntax

Add Volume *Parameters...*

Summary

Adds bar element volumes, by name, to a boundary condition's extent.

Parameter	Value	Default
Parameters	string...	–

Annulus Diameter Ratio Function

Syntax

Annulus Diameter Ratio Function *{=} Function*

Summary

Specifies the name of a user tabular function for annulus diameter ratio.

Parameter	Value	Default
{=}	{= | are | is}	–
Function	string	–

Annulus Diameter Ratio Threshold

Syntax

Annulus Diameter Ratio Threshold *{=}* *Threshold*

Summary

Specifies the minimum value for annulus diameter ratio. This is used with both automatic calculation as well as user tabular function.

Parameter	Value	Default
{=}	{= | are | is}	–
Threshold	real	0.05

Coordinate Reference

Syntax

Coordinate Reference *{=}* *Nodelist* Component *{s | x | y | z}*

Summary

Specifies the reference nodelist and direction for bar coordinates.

Parameter	Value	Default
{=}	{= | are | is}	–
Nodelist	string	–
xyzs	{s | x | y | z}	–

Entrance Effect Starting Node Id

Syntax

Entrance Effect Starting Node Id *{=}* *NodeID* [*Offset* *{=}* *Value*]

Summary

Specifies the node in the advective bar that corresponds to the position where entrance effects begin with pipe and annular flow correlations (Types 23, 25, 72-74).

Description

Pipe and annular flow correlations support an entrance effect correction factor that will adjust the local heat transfer coefficient based on how far a point is from the inlet of the pipe or annulus. By default the inlet of the pipe is assumed to be at the first node of the advective bar. This option allows the user to specify a

different node ID as the pipe entrance for calculating the entrance effect. This node ID must be part of the advective bar network.

Optionally, in cases where the entrance effect begins at a point not exactly aligned with the STARTING NODE one may specify an offset that will align the entrance effect to the correct coordinate. Here the offset is defined in the COORDINATE REFERENCE direction.

Parameter	Value	Default
{=}	{= | are | is}	–
NodeID	integer	0

Flow Cross Section Area

Syntax

Flow Cross Section Area *{=}* *Value*

Summary

Specifies a constant cross section area.

Parameter	Value	Default
{=}	{= | are | is}	–
Value	real	–

Flow Cross Sectional Area Function

Syntax

Flow Cross Sectional Area Function *{=}* *Function*

Summary

Specifies the name of a user tabular function for cross-sectional flow area.

Parameter	Value	Default
{=}	{= | are | is}	–
Function	string	–

Fluid Density

Syntax

Fluid Density *{=}* *Value*

Summary

Specifies bar fluid density.

Parameter	Value	Default
{=}	{= | are | is}	–
Value	real	–

Fluid Density Function

Syntax

Fluid Density Function *{=}* *Function*

Summary

Specifies the name of a user tabular function of temperature for density.

Parameter	Value	Default
{=}	{= | are | is}	–
Function	string	–

Hydraulic Diameter Function

Syntax

Hydraulic Diameter Function *{=}* *Function*

Summary

Specifies the name of a user tabular function for hydraulic diameter.

Parameter	Value	Default
{=}	{= | are | is}	–
Function	string	–

Initial Temperature

Syntax

Initial Temperature *{=}* *t*

Summary

Specifies the initial temperature of the bulk node.

Parameter	Value	Default
{=}	{= | are | is}	–
t	real	–

Mass Flow Rate

Syntax

Mass Flow Rate *{on}* *Nodelist* *{=}* *Value*

Summary

Specifies the mass flow rate at a node.

Parameter	Value	Default
{on}	{@ | at | for | in | on | over}	–
Nodelist	string	–
{=}	{= | are | is}	–
Value	real	–

Mass Flow Rate Function

Syntax

Mass Flow Rate Function *{on}* *Nodelist* *{=}* *Function*

Summary

Specifies the mass flow rate as a function of time at a node with a user function.

Parameter	Value	Default
{on}	{@ | at | for | in | on | over}	–
Nodelist	string	–
{=}	{= | are | is}	–
Function	string	–

Maximum Search Tolerance

Syntax

Maximum Search Tolerance *{=}* *Tolerance*

Summary

Specifies the maximum search tolerance for bar/facet geometric coupling search and this must be between 0 and 1. For an idealized case of a curved bar with a constant curvature (R) and a facet that is at a distance (d) from the bar, the analytical solution tolerance is given by

$$tol = \frac{d}{2R}$$

Note that for a straight bar with infinite radius of curvature, this gives a tolerance of 0.

Parameter	Value	Default
{=}	{= | are | is}	–
Tolerance	real	0.5

Search Interval

Syntax

Search Interval *{=}* *Interval*

Summary

Specifies the number of search intervals for bar/facet geometric coupling search. This is used in conjunction with the maximum search tolerance to generate search tolerances from 0.0 to the maximum search tolerance specified. Must be greater than 0

Parameter	Value	Default
{=}	{= | are | is}	–
Interval	integer	5

Velocity Bc

Syntax

Velocity Bc *{on}* *Nodelist* *{=}* *Velocity*

Summary

Specifies bar inlet velocity.

Parameter	Value	Default
{on}	{@ | at | for | in | on | over}	–
Nodelist	string	–
{=}	{= | are | is}	–
Velocity	real	–

Visualization

Syntax

Visualization *{off | on}*

Summary

Specifies Exodus output toggling of the created subsetted surfaces which are directly affected by the bulk nodes on the advective bar.

Parameter	Value	Default
OnOff	{off | on}	Off

Wetted Perimeter Function

Syntax

Wetted Perimeter Function *{=}* *Function*

Summary

Specifies the name of a user tabular function for wetted perimeter.

Parameter	Value	Default
{=}	{= | are | is}	–
Function	string	–

7.20 Bulk Nodes

Bulk Fluid Element

Scope

Aria Region, Equation System, Explicit Equation System, Root Finder Equation System

Summary

Defines a bulk fluid element that can be used by a convective flux boundary condition.

Description

Generally speaking, the reference temperature used in Newton's Law of Cooling and is a known quantity because the fluid with which it is associated is modeled as an infinite reservoir. However, if the size of this reservoir is finite, then its temperature can be affected by the energy transfer across the surface in question. This situation can be modeled by the bulk fluid element, wherein the energy of the reservoir is determined by a finite volume conservation equation.

For a bulk fluid element with a name specified as *name*, the thermodynamic properties such as density, volume and temperature may be accessed via global variables. Density is accessed using *name*_RHO, volume by *name*_V, temperature at the current state by *name*_T and temperature at the old state by *name*_TOld.

```
begin Bulk Fluid Element Name

   Bulk Coordinates {=} Parameters...

   Bulk Element Pressure {=} VolType [Parameters]...

   Bulk Element Volume {=} VolType [Parameters]...

   Bulk Eq EquationName For AssociatedDoF ([{of} SpeciesName]  | [{in}␣
↪MaterialPhaseName]) Using ElementType With Terms...

   Bulk Source For EquationName {=} SrcType [Parameters]...

   Calculate Volume From Enclosing Surface
```

```
   Initial Pressure {=} p

   Initial Temperature {=} t

   Material {=} MatName

   Reinitialize Dof Name {=} value

   Temperature Is Celsius

   Update Bulk Volume Pressure [ Atmospheric {=} Value  ]

   Use Block BlockName

end Bulk Fluid Element Name
```

Line Commands

Bulk Coordinates

Syntax

Bulk Coordinates *{=} Parameters. . .*

Summary

Specifies the physical coordinates for the bulk node.

Parameter	Value	Default
{=}	{= | are | is}	–
Parameters	real. . .	–

Bulk Element Pressure

Syntax

Bulk Element Pressure *{=} VolType [Parameters]. . .*

Summary

Specifies the pressure of the bulk element with type Pressure Type (CONSTANT, USER) for bulk element defined by the specifications of the parameter list.

This command should not be used when the bulk element is associated with a pressurization zone model since initial pressure will instead be associated with the pressurization zone as a whole.

Parameter	Value	Default
{=}	{= | are | is}	–
VolType	string	–
Parameters	[string]. . .	–

Bulk Element Volume

Syntax

Bulk Element Volume *{=} VolType [Parameters]. . .*

Summary

Specifies the volume of the bulk element with type VolType (CONSTANT, USER) for bulk element defined by the specifications of the parameter list.

For problems in which the mesh is deforming one should consider using CALCULATE VOLUME FROM ENCLOSING SURFACE instead.

Parameter	Value	Default
{=}	{= | are | is}	–
VolType	string	–
Parameters	[string]. . .	–

Bulk Eq

Syntax

Bulk Eq *EquationName* For *AssociatedDoF* ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*]) Using *ElementType* With *Terms. . .*

Summary

Specifies an equation to solve on the bulk node. User provides the degree of freedom associated with the equation, the element type, and the terms in the equation that are active.

Parameter	Value	Default
EquationName	string	–
AssociatedDoF	string	–
{of}	{of | species}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
ElementType	string	–
Terms	string. . .	–

Bulk Source For

Syntax

Bulk Source For *EquationName {=} SrcType [Parameters]. . .*

Summary

Indicates a source term of model type (i.e. CONSTANT, USER, CALORE_USER_SUB) and associated model Parameters for the given EquationName on the bulk element.

Parameter	Value	Default
EquationName	string	–
{=}	{= | are | is}	–
SrcType	string	–
Parameters	[string]. . .	–

Calculate Volume From Enclosing Surface

Syntax

Calculate Volume From Enclosing Surface

Summary

Specifies that the volume of the bulk element is to be calculated at each timestep from an enclosing surface. The surface must be closed (although Aria currently does not currently check for this) and is defined by the extent of the flux boundary condition which uses this bulk element.

Initial Pressure

Syntax

Initial Pressure *{=} p*

Summary

Specifies the initial pressure of the bulk node.

This command should not be used when the bulk element is associated with a pressurization zone model since initial pressure will instead be associated with the pressurization zone as a whole.

Parameter	Value	Default
{=}	{= | are | is}	–
p	real	0.0

Initial Temperature

Syntax

Initial Temperature *{=} t*

Summary

Specifies the initial temperature of the bulk node. If initial temperature values

are supplied from file this value will be ignored and file value is used instead. If one wishes to override the file value or if the file does not contain the bulk node use REINITIALIZE DOF.

Parameter	Value	Default
{=}	{= | are | is}	–
t	real	REAL_MAX

Material

Syntax

Material *{=}* *MatName*

Summary

Specifies that the material properties of the bulk element are defined in the material block named matName.

Parameter	Value	Default
{=}	{= | are | is}	–
MatName	string	–

Reinitialize Dof

Syntax

Reinitialize Dof *Name* *{=}* *value*

Summary

Requests reinitialization of a bulk node DOF. This will allow override of the file supplied bulk node DOF value when using an initial condition from file. Here the initial value supplied from file will be replaced by a specified value. Likewise, if the initial condition file does not contain the bulk node this value is assigned as the initial condition.

Parameter	Value	Default
Name	string	–
{=}	{= | are | is}	–
value	real	–

Temperature Is Celsius

Syntax

Temperature Is Celsius

Summary

Specifies the initial temperature of the bulk node is given in degrees Celsius.

Internal to the code this is relevant only when performing updates of the bulk element pressure via an ideal gas law which requires an absolute temperature.

Update Bulk Volume Pressure

Syntax

Update Bulk Volume Pressure [*Atmospheric {=} Value*]

Summary

Reevaluate non-pressurization zone model bulk volume pressure.

Description

If a bulk volume is filled with an ideal gas then the pressure can be calculated at each timestep provided that the initial temperature and pressure are given. Here it is assumed that the associated volume is closed and that the volume has been defined as constant or is being computed. If the bulk volume is being computed then one must also include the CALCULATE VOLUME FROM ENCLOSING SURFACE command line.

By default the resulting update is an absolute pressure. If one wishes the update to reflect gauge pressure one must supply a reference pressure by adding an optional Atmospheric = value syntax to the command line.

This command should not be used when the bulk element is associated with a pressurization zone model since pressure is instead associated with the pressurization zone as a whole. The pressurization zone model also allows for gas models other than ideal gas.

Use Block

Syntax

Use Block *BlockName*

Summary

Specifies that the given element block should be used for the bulk node. The caveat here is that there must be only one element, one node and they must all satisfy the proper mesh object topologies. This option is used only if an element block for the bulk element is already present in the mesh. If the bulk element does not exist on the input mesh, the element block will be created and will exist on the output mesh.

Parameter	Value	Default
BlockName	string	–

Bulk Nodes Ignore Restart

Syntax

Bulk Nodes Ignore Restart

Summary

Causes bulk nodes to ignore the restart file.

Bulk Node Coupling

Scope

Aria Region, Equation System, Explicit Equation System, Root Finder Equation System

Summary

Defines a coupling between equations on two bulk nodes.

Description

This block command allows a user to couple equations on different bulk nodes to one another. For example, the density of gas on two bulk nodes can be coupled using a flow coefficient based on the relative pressure of the two bulk nodes. The same degree of freedom must be present on both bulk nodes, and both have to be in the same equation system in order for the coupling to be possible.

```
begin Bulk Node Coupling Name

   Additional Parameter PropertyName {=} ModelName [Parameters]...

   Bulk Nodes {=} Names1 Names2...

   Couple DofName Model {=} ModelName [Parameters]...

end Bulk Node Coupling Name
```

Line Commands

Additional Parameter

Syntax

Additional Parameter *PropertyName {=} ModelName [Parameters]...*

Summary

Declare an additional parameter for the coupling models present. For example, the "K_FACTOR_FLOW" model for density or species coupling requires that a "K_FACTOR" be provided using this line command.

Parameter	Value	Default
PropertyName	string	–
{=}	{= | are | is}	–
ModelName	string	–
Parameters	[string]. . .	–

Bulk Nodes

Syntax

Bulk Nodes *{=} Names1 Names2. . .*

Summary

Specify the name of the bulk nodes involved in the coupling.

Parameter	Value	Default
{=}	{= | are | is}	–
Names	string1 string2. . .	–

Couple

Syntax

Couple *DofName* Model *{=} ModelName [Parameters]. . .*

Summary

Specify what degree of freedom to couple between the bulk nodes, and what model to use for the coupling.

Parameter	Value	Default
DofName	string	–
{=}	{= | are | is}	–
ModelName	string	–
Parameters	[string]. . .	–

7.21 CDFEM Options

Cdfem Options

Scope

Aria Region

Summary

Main block for setting CDFEM options.

Description

This block is where CDFEM options that apply to all CDFEM interfaces in a problem (level sets, CDFEM death, CDFEM isosurfaces, and CDFEM irreversible phase change) can be set. These options include the edge degeneracy handling method, edge tolerance, prolongation model, etc.

begin Cdfem Options

Cdfem Dof Edge Tolerance *{=}* *Value*

Cdfem Edge Degeneracy Handling *{=}* *{snap_to_interface_when_quality_allows_then_snap_to_node | snap_to_node}*

Cdfem Edge Tolerance *{=}* *Value*

Cdfem Element Size *{=}* *{conformal | nonconformal}*

Cdfem Internal Side Stabilization [Multiplier *{=}* *Multiplier*]

Cdfem Nonconformal Adaptivity Levels *{=}* *Value*

Cdfem Prolongation Model *{=}* *{ale_closest_node | ale_closest_point | ale_nearest_node | ale_nearest_point | interpolation}*

Cdfem Simplex Generation Method *{=}* *{cut_quads_by_global_identifier | cut_quads_by_largest_angle, | cut_quads_by_nearest_edge_cut}*

Constrain Cdfem To Xfem Space *{=}* *{false | no | off | on | true | yes}*

Initial Prolongation Field For *DestFieldName* *{=}* *SrcFieldName*

Length Scale For Interface Courant Limit *{=}* *{constant | l1_norm | local}* [Value *{=}* *ConstantLengthScaleVelue*]

Levelset Refinement Interval *{=}* *interval1 interval2*

Max Edge Snap *{=}* *value*

Nonconformal Adaptivity Target Element Count *{=}* *Value*

Number Of Initial Decomposition Cycles *{=}* *Value*

Perform Volume Correction After Levelset Solve *{=}* *{false | no | off | on | true | yes}*

```
  Preserve Sharp Features When Snapping [ Feature_Angle {=} Angle  ]

  Prolongate Additional Fields {=} FieldNames...

  Resnap Mesh Using Interface On Previous Snapped Mesh {=} {false | no |␣
↪off | on | true | yes}

  Resnap Mesh Using Interpolation {=} {false | no | off | on | true |␣
↪yes}

  Use Ale Prolongation For FieldName

  Use Hierarchical Dofs {=} {false | no | off | on | true | yes}

  Use Interpolation To Unsnap Mesh {=} {false | no | off | on | true |␣
↪yes}

  Use Velocity To Evaluate Interface Courant Limit {=} {false | no | off␣
↪| on | true | yes}

end Cdfem Options
```

Line Commands

Cdfem Dof Edge Tolerance

Syntax

Cdfem Dof Edge Tolerance *{=}* *Value*

Summary

Specify the tolerance used to "snap" the DOFs to the edge node values on CDFEM meshes.

Parameter	Value	Default
{=}	{= | are | is}	–
Value	real	0.0

Cdfem Edge Degeneracy Handling

Syntax

Cdfem Edge Degeneracy Handling *{=}*
{snap_to_interface_when_quality_allows_then_snap_to_node | snap_to_node}

Summary

Specify the strategy for handling edges that are cut very close to a node.

Parameter	Value	Default
{=}	{= | are | is}	–
CDFEMEdgeDegeneracyHandlingType	{snap_to_interface_when_quality_allows_then_snap_to_node | snap_to_node}	GLOBAL_SNAP_TO_NODE

Cdfem Edge Tolerance

Syntax

Cdfem Edge Tolerance *{=}* *Value*

Summary

Specify the tolerance used to "snap" the CDFEM conformal interface to the nonconformal mesh.

Parameter	Value	Default
{=}	{= | are | is}	–
Value	real	1.0e-3

Cdfem Element Size

Syntax

Cdfem Element Size *{=}* *{conformal \| nonconformal}*

Summary

Specify whether the element size is based on CONFORMAL OR NONCONFORMAL element (used for pressure stabilization).

Parameter	Value	Default
{=}	{= | are | is}	–
CDFEMElementSize	{conformal | nonconformal}	NONCONFORMAL

Cdfem Internal Side Stabilization

Syntax

Cdfem Internal Side Stabilization [Multiplier *{=}* *Multiplier*]

Summary

Turn on internal face stabilization on equations that support it, optionally specifying a multiplier (default = 1.0).

Cdfem Nonconformal Adaptivity Levels

Syntax

Cdfem Nonconformal Adaptivity Levels *{=}* *Value*

Summary

Specify the number of levels of non-conformal adaptivity that should be

performed before the conformal decomposition is performed.

Parameter	Value	Default
{=}	{= | are | is}	–
Value	integer	0

Cdfem Prolongation Model

Syntax

Cdfem Prolongation Model *{=} {ale_closest_node | ale_closest_point | ale_nearest_node | ale_nearest_point | interpolation}*

Summary

Specify the strategy for prolonging nodes created by conformal decomposition.

Parameter	Value	Default
{=}	{= | are | is}	–
CDFEMEProlongationModelType	{ale_closest_node | ale_closest_point | ale_nearest_node | ale_nearest_point | interpolation}	ALE_CLOSEST_POINT

Cdfem Simplex Generation Method

Syntax

Cdfem Simplex Generation Method *{=} {cut_quads_by_global_identifier | cut_quads_by_largest_angle, | cut_quads_by_nearest_edge_cut}*

Summary

Specify how the quadrilateral elements (2D) or quadrilateral faces (3D) that are generated by the conformal decomposition process are subdivided into triangles. Default: For 2D: CUT_QUADS_BY_LARGEST_ANGLE, For 3D: CUT_QUADS_BY_NEAREST_EDGE_CUT

Parameter	Value	Default
{=}	{= | are | is}	–
CDFEMSimplexGenerationMethod	{cut_quads_by_global_identifier | cut_quads_by_largest_angle, | cut_quads_by_nearest_edge_cut}	–

Constrain Cdfem To Xfem Space

Syntax

Constrain Cdfem To Xfem Space *{=} {false | no | off | on | true | yes}*

Summary

Specify whether the CDFEM DOFs should be constrained to the coarser XFEM discrete space. Default = FALSE.

Parameter	Value	Default
{=}	{= | are | is}	–
constrainCDFEMtoXFEM	{false | no | off | on | true | yes}	FALSE

Initial Prolongation Field For

Syntax

Initial Prolongation Field For *DestFieldName {=} SrcFieldName*

Summary

Specifies that the field DestFieldName will use the field SrcFieldName for prolongation if (no prolongation node is found OR DestFieldName is not defined on the prolongation node) AND DestFieldName was not defined on the node at the last time step. For example this could be used in a foam decomposition problem where no fluid region is initially present, but one appears as the foam decomposes and the fluid field values should be populated based on the foam field values, e.g. INITIAL PROLONGATION FIELD FOR solution->pressure is solution->gas_phase_pressure

Parameter	Value	Default
DestFieldName	string	–
{=}	{= | are | is}	–
SrcFieldName	string	–

Length Scale For Interface Courant Limit

Syntax

Length Scale For Interface Courant Limit *{=} {constant | l1_norm | local}* [Value *{=} ConstantLengthScaleVelue*]

Summary

Specify how the length scale is computed in the Interface CFL calculation. If CONSTANT is specified then the value is also given via VALUE=NNN. If L1_NORM is specified, the average element size across the conforming interfaces is used. Default is LOCAL, meaning that the local element size is used.

Parameter	Value	Default
{=}	{= | are | is}	–
InterfaceCFLLengthScaleType	{constant | l1_norm | local}	–

Levelset Refinement Interval

Syntax

Levelset Refinement Interval *{=}* *interval1 interval2*

Summary

Use this command to specify the range of level set values that should be fully refined. If this is not specified, the elements that span the zero level set are refined.

Parameter	Value	Default
{=}	{= | are | is}	–
interval	real1 real2	–

Max Edge Snap

Syntax

Max Edge Snap *{=}* *value*

Summary

Use this command to specify the largest relative snap that should be performed on cut edges. If this is not specified, any size edge snap is allowed (default = 1.0).

Parameter	Value	Default
{=}	{= | are | is}	–
value	real	–

Nonconformal Adaptivity Target Element Count

Syntax

Nonconformal Adaptivity Target Element Count *{=}* *Value*

Summary

Specify a target element count for the nonconformal adaptivity. This will cause the non-conformal adaptivity to use at most "CDFEM NONCONFORMAL ADAPTIVITY LEVELS" levels of refinement of a given parent element and will only refine as elements until the target element count is reached instead of refining all elements cut by the interface. At present this capability only works with analytic initial conditions in krino.

Parameter	Value	Default
{=}	{= | are | is}	–
Value	real	0

Number Of Initial Decomposition Cycles

Syntax

Number Of Initial Decomposition Cycles *{=}* *Value*

Summary

Specify the number of times the mesh is decomposed and the fields are (re)initialized. Repeated decomposition and initialization improves the accuracy of the interface location because the added nodes are also initialized.

Parameter	Value	Default
{=}	{= | are | is}	–
Value	integer	1

Perform Volume Correction After Levelset Solve

Syntax

Perform Volume Correction After Levelset Solve *{=}* *{false | no | off | on | true | yes}*

Summary

Use this command to specify whether interpolation should be used to unsnap the mesh. Default is false (in which case ALE is used in aria).

Parameter	Value	Default
{=}	{= | are | is}	–
flag	{false | no | off | on | true | yes}	–

Preserve Sharp Features When Snapping

Syntax

Preserve Sharp Features When Snapping [Feature_Angle *{=}* *Angle*]

Summary

When snapping the mesh to the interface, respect sharp features, optionally specifying the FEATURE ANGLE in degrees (default = 135.).

Prolongate Additional Fields

Syntax

Prolongate Additional Fields *{=}* *FieldNames. . .*

Summary

Use this command to add additional fields to be prolongated in CDFEM problems (this prolongation is a nearest point projection).

Parameter	Value	Default
{=}	{= | are | is}	–
FieldNames	string. . .	–

Resnap Mesh Using Interface On Previous Snapped Mesh

Syntax

Resnap Mesh Using Interface On Previous Snapped Mesh *{=} {false | no | off | on | true | yes}*

Summary

Use this command to specify whether mesh resnapped using interface location on previous snapped mesh. Default is false (in which case ALE unsnap or INTERPOLATION unsnap is used instead in aria).

Parameter	Value	Default
{=}	{= | are | is}	–
flag	{false | no | off | on | true | yes}	–

Resnap Mesh Using Interpolation

Syntax

Resnap Mesh Using Interpolation *{=} {false | no | off | on | true | yes}*

Summary

Use this command to specify whether mesh resnapped using interpolation from the solution on the previous snapped mesh. Default is currently false (in which case ALE unsnap is used instead in aria).

Parameter	Value	Default
{=}	{= | are | is}	–
flag	{false | no | off | on | true | yes}	–

Use Ale Prolongation For

Syntax

Use Ale Prolongation For *FieldName*

Summary

WARNING: Use cases for this line command are VERY rare.

Force the use of ALE prolongation for the field specified by FieldName.

Parameter	Value	Default
FieldName	string	–

Use Hierarchical Dofs

Syntax

Use Hierarchical Dofs *{=} {false | no | off | on | true | yes}*

Summary

Specify whether the CDFEM DOFs should be transformed to hierarchical unknowns. This can significantly improve the performance of iterative solvers, especially in parallel. Default = FALSE.

Parameter	Value	Default
{=}	{= | are | is}	–
useHierarchicalDofs	{false | no | off | on | true | yes}	FALSE

Use Interpolation To Unsnap Mesh

Syntax

Use Interpolation To Unsnap Mesh *{=} {false | no | off | on | true | yes}*

Summary

Use this command to specify whether interpolation should be used to unsnap the mesh. Default is false (in which case ALE is used in aria).

Parameter	Value	Default
{=}	{= | are | is}	–
flag	{false | no | off | on | true | yes}	–

Use Velocity To Evaluate Interface Courant Limit

Syntax

Use Velocity To Evaluate Interface Courant Limit *{=} {false | no | off | on | true | yes}*

Summary

Specify whether the Interface Courant Limit should be calculated using the velocity on the interface. If true, the CFL is calculated using the normal of solution->Velocity field on the CDFEM interfaces. If false, the CFL is calculated using the CDFEM Mesh Displacements on the CDFEM interfaces. Default = FALSE.

Parameter	Value	Default
{=}	{= | are | is}	–
useVelocityToEvaluateInterfaceCFL	{false | no | off | on | true | yes}	FALSE

7.22 CHEMEQ

Parameters For Chemeq Model

Scope

Aria Material, Material Phase

Summary

Parameters for the CHEMEQ chemistry model within Aria. The ModelName must appear in a SRC term for an ENERGY equation in order to be applied.

begin Parameters For Chemeq Model *ModelName*

Activation Energies *{=} activation energies...*

Activation Energy St Devs *{=} activation energy st devs...*

Aux Variable Names *{=} aux variable names...*

Aux Variable Subroutine *{=} aux variable subroutine*

Concentration Exponents For *SpeciesName {=} Exponents...*

Concentration Units *{=} concentration units...*

Condensed Fraction *{=} condensed fraction*

Energy Release Units *{=} {per unit mass | per unit volume}*

Energy Releases *{=} energy releases...*

Enthalpies Of Formation *{=} enthalpies of formation...*

Extent Of Reaction Based On *extent of reaction based on...*

Log Preexponential Factors *{=} log preexponential factors...*

Number Of Reactions *{=} number of reactions*

Pressure *{=} pressure...*

Pressure Exponents *{=} pressure exponents...*

Rate Multiplier *{=} rate multiplier...*

```
Reaction Rate Model {=} reaction rate model

Reaction Rate Subroutine {=} reaction rate subroutine

Reference Pressure {=} reference pressure

Species Molecular Weights {=} species molecular weights...

Species Names {=} species names...

Species Phases {=} species phases...

Steric Coefficients {=} steric coefficients...

Stoichiometric Coefficients For SpeciesName {=} Exponents...

begin Reaction ReactionName
end
```

end Parameters For Chemeq Model *ModelName*

Line Commands

Activation Energies

Syntax

Activation Energies *{=} activation energies...*

Summary

This command specifies the activation energies for each reaction.

Parameter	Value	Default
{=}	{= | are | is}	–
activation energies	real...	–

Activation Energy St Devs

Syntax

Activation Energy St Devs *{=} activation energy st devs...*

Summary

This command supplies the standard deviations for the activation energies for each reaction. The order is significant, and corresponds to the order in which the Reactions are specified.

Parameter	Value	Default
{=}	{= | are | is}	–
activation energy st devs	real. . .	–

Aux Variable Names

Syntax

Aux Variable Names *{=} aux variable names. . .*

Summary

Supplies the names for any auxiliary variables. The number of auxiliary variables is implied by the size of this list.

Description

Warning: This command is deprecated and should not be used any more.

The entries must be separated by whitespace, without commas. These variables do not participate in the reaction kinetics, but are derived from the species variables and calculated in the user-defined subroutine named via the "AUX VARIABLE SUBROUTINE NAME" line command.

Parameter	Value	Default
{=}	{= | are | is}	–
aux variable names	string. . .	–

Aux Variable Subroutine

Syntax

Aux Variable Subroutine *{=} aux variable subroutine*

Summary

Supplies the name of the user-defined subroutine that performs any auxiliary variable calculations. This command is required if auxiliary variables are named.

Description

Warning: This command is deprecated and should not be used any more.

Parameter	Value	Default
{=}	{= | are | is}	–
aux variable subroutine	string	–

Concentration Exponents For

Syntax

Concentration Exponents For *SpeciesName {=} Exponents. . .*

Summary

For the given species, this command specifies the concentration exponents for each reaction.

Parameter	Value	Default
SpeciesName	string	–
{=}	{= | are | is}	–
Exponents	real. . .	–

Concentration Units

Syntax

Concentration Units *{=} concentration units. . .*

Summary

Specifies the concentration units.

Description

Specifies the concentration units. This input is required when using a pressurization model or using the `FROM_CHEMEQ` model.

Valid inputs for the four concentration types include:

- `Mass Fraction` or `Mass Fractions`
- `Mole Fraction` or `Mole Fractions`
- `Mass` or `Density`
- `Moles` or `Molar`

Parameter	Value	Default
{=}	{= | are | is}	–
concentration units	string. . .	UNKNOWN_UNITS

Condensed Fraction

Syntax

Condensed Fraction *{=} condensed fraction*

Summary

Specifies mass fraction of the condensed phase.

Parameter	Value	Default
{=}	{= | are | is}	–
condensed fraction	real	–

Energy Release Units

Syntax

Energy Release Units *{=} {per unit mass | per unit volume}*

Summary

Specifies the energy release units for each reaction, PER VOLUME or PER UNIT MASS.

Default is PER VOLUME. If PER UNIT MASS is selected, the specified heat release value is multiplied by the local density.

Parameter	Value	Default
{=}	{= | are | is}	–
energy release units	{per unit mass | per unit volume}	PER VOLUME

Energy Releases

Syntax

Energy Releases *{=} energy releases. . .*

Summary

This command specifies the energy release for each reaction.

Parameter	Value	Default
{=}	{= | are | is}	–
energy releases	real. . .	–

Enthalpies Of Formation

Syntax

Enthalpies Of Formation *{=} enthalpies of formation. . .*

Summary

This command specifies the enthalpies of formation for each species

Parameter	Value	Default
{=}	{= | are | is}	–
enthalpies of formation	real. . .	–

Extent Of Reaction Based On

Syntax

Extent Of Reaction Based On *extent of reaction based on. . .*

Summary

This command supplies the species on which the progress variable for each distributed activation energy depends. The order is significant, and corresponds to the order in which the Reactions are specified.

Parameter	Value	Default
extent of reaction based on	string. . .	–

Log Preexponential Factors

Syntax

Log Preexponential Factors *{=}* *log preexponential factors. . .*

Summary

This command specifies the natural logarithms of the pre-exponential factors for each reaction.

Parameter	Value	Default
{=}	{= | are | is}	–
log preexponential factors	real. . .	–

Number Of Reactions

Syntax

Number Of Reactions *{=}* *number of reactions*

Summary

Specifies the number of reactions for this material.

Parameter	Value	Default
{=}	{= | are | is}	–
number of reactions	integer	–

Pressure

Syntax

Pressure *{=}* *pressure. . .*

Summary

Specifies the expression for pressure for this material.

Description

Specifies the expression for pressure for this material. The preferred option is to

use `Pressure = From_Material_Definition` and put a pressure specification in the material block (e.g. `Pressure = Pressurization_Model`). If you are using pressurization zones, you must use this approach. Using `Pressure = Pressurization_Model` here will generate an error.

Parameter	Value	Default
{=}	{= | are | is}	–
pressure	string. . .	–

Pressure Exponents

Syntax

Pressure Exponents *{=} pressure exponents. . .*

Summary

This command specifies the pressure exponents for each reaction.

Parameter	Value	Default
{=}	{= | are | is}	–
pressure exponents	real. . .	–

Rate Multiplier

Syntax

Rate Multiplier *{=} rate multiplier. . .*

Summary

Specifies a generic expression for the rate multiplier for this ChemEq model. This input is optional, and the rate multiplier defaults to 1.0 if this is omitted.

Parameter	Value	Default
{=}	{= | are | is}	–
rate multiplier	string. . .	1

Reaction Rate Model

Syntax

Reaction Rate Model *{=} reaction rate model*

Summary

By default, reaction rates are Arrhenius. If this line command is included, then a named reaction rate model can be used. Current options are: “Prout_Tompkins”

Description

Warning: This command is deprecated and should not be used any more.

Parameter	Value	Default
{=}	{= | are | is}	–
reaction rate model	string	–

Reaction Rate Subroutine

Syntax

Reaction Rate Subroutine *{=}* *reaction rate subroutine*

Summary

By default, reaction rates are Arrhenius. If this line command is included, then the named subroutine is called instead.

Description

Warning: This command is deprecated and should not be used any more.

Parameter	Value	Default
{=}	{= | are | is}	–
reaction rate subroutine	string	–

Reference Pressure

Syntax

Reference Pressure *{=}* *reference pressure*

Summary

Specifies the reference pressure for this material.

Parameter	Value	Default
{=}	{= | are | is}	–
reference pressure	real	–

Species Molecular Weights

Syntax

Species Molecular Weights *{=}* *species molecular weights. . .*

Summary

Supplies the molecular weight for each chemical species. The order is significant, and corresponds to the order in which the Species Names are specified.

Parameter	Value	Default
{=}	{= | are | is}	–
species molecular weights	real. . .	–

Species Names

Syntax

Species Names *{=} species names. . .*

Summary

Supplies names for the chemical species. The number of chemical species is implied by the size of this list. The entries must be separated by whitespace, without commas.

Parameter	Value	Default
{=}	{= | are | is}	–
species names	string. . .	–

Species Phases

Syntax

Species Phases *{=} species phases. . .*

Summary

Supplies the phase for each chemical species. The order is significant, and corresponds to the order in which the Species Names are specified.

Parameter	Value	Default
{=}	{= | are | is}	–
species phases	{condensed | gas}	–

Steric Coefficients

Syntax

Steric Coefficients *{=} steric coefficients. . .*

Summary

This command specifies the steric coefficients for each reaction.

Parameter	Value	Default
{=}	{= | are | is}	–
steric coefficients	real. . .	–

Stoichiometric Coefficients For

Syntax

Stoichiometric Coefficients For *SpeciesName {=} Exponents. . .*

Summary

For the given species, this command specifies the stoichiometric coefficients for each reaction.

Parameter	Value	Default
SpeciesName	string	–
{=}	{= | are | is}	–
Exponents	real. . .	–

Chemeq Solver For

Scope

Aria Region, Equation System, Explicit Equation System, Root Finder Equation System

Summary

Parameters for the CHEMEQ solver applicable to the previously defined material with CHEMEQ model ModelName. The CHEMEQ SOLVER model produces no independent action since the ModelName must first appear in a SRC term for an ENERGY equation before it be applied.

```
begin Chemeq Solver For ModelName

   Absolute Tolerance {=} absTol

   Activation Temperature {=} T_act

   Activation Temperature Rate {=} T_dot

   Activation Time {=} T_init

   Allow Old Heat Generation Method {=} {false | no | off | on | true |␣
↪yes}

   Asymptotic Tolerance {=} Tol
```

```
   Aux Variable Name {=} C_init
   Chemistry Step Multiplier {=} Dt_cond_over_dt_chem
   Deactivation Expression {=} UserTag {continue | terminate}
   Deactivation Temperature {=} t {continue | terminate}
   Deactivation Temperature Overshoot {=} dT_overshoot
   Deactivation Temperature Rate {=} T_dot {continue | terminate}
   Deactivation Time {=} T_stop {continue | terminate}
   Enable Enthalpy Integration
   Epsilon Max {=} E_max
   Epsilon Min {=} E_min
   Maximum Substeps {=} maxSubSteps
   Minimum Chemistry Timestep {=} Dt_min
   Minimum Concentration For Name {=} C_min
   Ode Solver {=} odeSolver...
   Percentage Asymptotics {=} PercentValue
   Post Deactivation Gas Release {=} gasRelease
   Post Deactivation Heat Release {=} heatRelease
   Post Deactivation Release Time {=} releaseTime
   Rate Limiter Factor {=} rateLimiterFactor
   Relative Tolerance {=} relTol
   Species Name {=} Conc
end Chemeq Solver For ModelName
```

Line Commands

Absolute Tolerance

Syntax

Absolute Tolerance *{=}* *absTol*

Summary

Specifies the absolute solution tolerance for the ODE solver

Parameter	Value	Default
{=}	{= | are | is}	–
absTol	real	1e-12

Activation Temperature

Syntax

Activation Temperature *{=}* *T_act*

Summary

Specification of threshold activation temperature for chemistry.

Parameter	Value	Default
{=}	{= | are | is}	–
T_act	real	0.0

Activation Temperature Rate

Syntax

Activation Temperature Rate *{=}* *T_dot*

Summary

Specification of temperature rate limit (dT/dt) for which chemistry activation occurs.

Parameter	Value	Default
{=}	{= | are | is}	–
T_dot	real	REAL_MAX

Activation Time

Syntax

Activation Time *{=}* *T_init*

Summary

Specification of activation time for chemistry.

Parameter	Value	Default
{=}	{= | are | is}	–
T_init	real	REAL_MAX

Allow Old Heat Generation Method

Syntax

Allow Old Heat Generation Method *{=} {false | no | off | on | true | yes}*

Summary

This command has been removed, using it will result in an error.

Description

Warning: This command is deprecated and should not be used any more.

Parameter	Value	Default
{=}	{= | are | is}	–
Bool	{false | no | off | on | true | yes}	FALSE

Asymptotic Tolerance

Syntax

Asymptotic Tolerance *{=} Tol*

Summary

Specifies the asymptotic selection criterion for the chemical rate equations.

Description

Warning: This command is deprecated and should not be used any more.

Parameter	Value	Default
{=}	{= | are | is}	–
Tol	real	100.0

Aux Variable

Syntax

Aux Variable *Name {=} C_init*

Summary

Specifies the initial value for the named auxiliary variable.

Description

Warning: This command is deprecated and should not be used any more.

Parameter	Value	Default
Name	string	–
{=}	{= | are | is}	–
C_init	real	0.0

Chemistry Step Multiplier

Syntax

Chemistry Step Multiplier *{=}* *Dt_cond_over_dt_chem*

Summary

This command specifies the ratio of the maximum allowed conduction timestep to the smallest, current chemistry time step.

Description

Warning: This command is deprecated and should not be used any more.

Parameter	Value	Default
{=}	{= | are | is}	–
Dt_cond_over_dt_chem	real	100.0

Deactivation Expression

Syntax

Deactivation Expression *{=}* *UserTag* *{continue | terminate}*

Summary

Specification of deactivation criteria based on a user expression within the material model and how the simulation should proceed after deactivation. The supplied user expression will automatically have MAX and VALUE postprocess fields associated with it.

Parameter	Value	Default
{=}	{= | are | is}	–
UserTag	"string"	–
ChemistryDeactivationAction	{continue | terminate}	TERMINATE

Deactivation Temperature

Syntax

Deactivation Temperature *{=} t {continue | terminate}*

Summary

Specification of deactivation temperature and how the simulation should proceed after deactivation.

Parameter	Value	Default
{=}	{= | are | is}	–
t	real	REAL_MAX
ChemistryDeactivationAction	{continue | terminate}	TERMINATE

Deactivation Temperature Overshoot

Syntax

Deactivation Temperature Overshoot *{=} dT_overshoot*

Summary

Specification of acceptable overshoot of the deactivation temperature for the limiting procedure

Parameter	Value	Default
{=}	{= | are | is}	–
dT_overshoot	real	1.0

Deactivation Temperature Rate

Syntax

Deactivation Temperature Rate *{=} T_dot {continue | terminate}*

Summary

Specification of temperature rate limit (T_dot) for deactivation and how the simulation should proceed after deactivation.

Parameter	Value	Default
{=}	{= | are | is}	–
T_dot	real	REAL_MAX
ChemistryDeactivationAction	{continue | terminate}	TERMINATE

Deactivation Time

Syntax

Deactivation Time *{=}* *T_stop* *{continue | terminate}*

Summary

Specification of deactivation time for chemistry and how the simulation should proceed after deactivation.

Parameter	Value	Default
{=}	{= | are | is}	–
T_stop	real	REAL_MAX
ChemistryDeactivationAction	{continue | terminate}	TERMINATE

Enable Enthalpy Integration

Syntax

Enable Enthalpy Integration

Summary

This command has been removed, using it will result in an error.

Description

Warning: This command is deprecated and should not be used any more.

Epsilon Max

Syntax

Epsilon Max *{=}* *E_max*

Summary

Specifies the upper bound convergence criterion and chemistry time scale check.

Description

Warning: This command is deprecated and should not be used any more.

Parameter	Value	Default
{=}	{= | are | is}	–
E_max	real	10.0

Epsilon Min

Syntax

Epsilon Min *{=}* *E_min*

Summary

Specifies the lower bound convergence criterion and chemistry time scale check.

Description

Warning: This command is deprecated and should not be used any more.

Parameter	Value	Default
{=}	{= | are | is}	–
E_min	real	0.0001

Maximum Substeps

Syntax

Maximum Substeps *{=}* *maxSubSteps*

Summary

Specifies the maximum number of chemistry substeps per global time step.

Parameter	Value	Default
{=}	{= | are | is}	–
maxSubSteps	integer	100000

Minimum Chemistry Timestep

Syntax

Minimum Chemistry Timestep *{=}* *Dt_min*

Summary

Specifies the minimum time step allowed during the solution of the chemical rate equations.

Parameter	Value	Default
{=}	{= | are | is}	–
Dt_min	real	1.0e-15

Minimum Concentration For

Syntax

Minimum Concentration For *Name* *{=}* *C_min*

Summary

Specifies the minimum concentration for the named species.

Description

Warning: This command is deprecated and should not be used any more.

Parameter	Value	Default
Name	string	–
{=}	{= | are | is}	–
C_min	real	1.0e-12

Ode Solver

Syntax

Ode Solver *{=}* *odeSolver. . .*

Summary

This command selects the ODE solver to use for the CHEMEQ solve.

Description

For available options, refer to the ‘ODE Solvers’ section in the ‘Chemistry’ chapter of the user manual. This command is required to activate an ODE solver.

Parameter	Value	Default
{=}	{= | are | is}	–
odeSolver	string. . .	NONE

Percentage Asymptotics

Syntax

Percentage Asymptotics *{=}* *PercentValue*

Summary

Specifies the percentage of the chemical rate equations that will always be solved using asymptotics.

Description

Warning: This command is deprecated and should not be used any more.

Parameter	Value	Default
{=}	{= | are | is}	–
PercentValue	real	0.0

Post Deactivation Gas Release

Syntax

Post Deactivation Gas Release *{=}* *gasRelease*

Summary

Amount of gas to be released uniformly after deactivation (units are always in moles per volume).

Parameter	Value	Default
{=}	{= | are | is}	–
gasRelease	real	0.01

Post Deactivation Heat Release

Syntax

Post Deactivation Heat Release *{=}* *heatRelease*

Summary

Amount of energy to be released uniformly after deactivation. The units must be consistent with the base ChemEq heat release units (energy per volume or energy per mass).

Parameter	Value	Default
{=}	{= | are | is}	–
heatRelease	real	0.01

Post Deactivation Release Time

Syntax

Post Deactivation Release Time *{=}* *releaseTime*

Summary

Time after deactivation for constant heat and gas sources to be active

Parameter	Value	Default
{=}	{= | are | is}	–
releaseTime	real	0.01

Rate Limiter Factor

Syntax

Rate Limiter Factor *{=}* *rateLimiterFactor*

Summary

Enables a rate limiter and specifies what fraction of the fluid time step to consider for the limiter time step.

Parameter	Value	Default
{=}	{= | are | is}	–
rateLimiterFactor	real	0.01

Relative Tolerance

Syntax

Relative Tolerance *{=}* *relTol*

Summary

Specifies the relative solution tolerance for the ODE solver

Parameter	Value	Default
{=}	{= | are | is}	–
relTol	real	1e-6

Species

Syntax

Species *Name* *{=}* *Conc*

Summary

Specifies the initial concentration for the named species.

Parameter	Value	Default
Name	string	–
{=}	{= | are | is}	–
Conc	real	0.0

7.23 Contact

Table of Contents

- *Contact Definition*
- *Interaction*

- *Interaction Defaults*
- *Search Options*
- *Enforcement*

Contact Definition

Scope

Aria Region, Equation System, Explicit Equation System, Root Finder Equation System

Summary

Contains the commands needed to define contact for an analysis.

Description

This block command defines a scenario for the contact model. The contact surfaces are defined, the interactions are set, and the type of contact enforcement is defined.

```
begin Contact Definition Contact_name

  Contact Surface Surface_name [ Contains List_of_instances...  ]

  Disable Opposed Normal Interaction Culling

  Opposed Normal Threshold Angle {=} Angle

  Output Rule {=} {more verbose | none | summary | verbose}

  Periodicity For surface {=} Rectangular Offset = offset1 offset2[␣
↪offset3]

  Periodicity For surface {=} Align_Surface (Destination = destination ␣
↪| [Invert_Normals = invert_normals])

  Periodicity For surface {=} Cylindrical ([Axis = axis1 axis2[ axis3]] ␣
↪| [Center = center1 center2[ center3]]  | Angle = angle)

  Periodicity For surface {=} Mapped ([X_Map = x_map]  | [Y_Map = y_map]␣
↪ | [Z_Map = z_map])

  Restrict Contact To Subdomain Defined By Radius r And Point VecX VecY␣
↪VecZ
```

```
Skin All Blocks {=} {off | on} [ Exclude Blocks_to_exlude...  ]

Update Search Every n Steps

Visualize Contact

begin Enforcement Enforcement_name
end

begin Interaction Interaction_name
end

begin Interaction Defaults
end

begin Search Options Search_options_name
end

end Contact Definition Contact_name
```

Line Commands

Contact Surface

Syntax

Contact Surface *Surface_name* [Contains *List_of_instances...*]

Summary

Defines a surface made up of a set of surfaces in the mesh file to consider for contact. Specification of an element block implies that block skinning of the element block is to take place. The specific implementation of block skinning depends on the application.

Parameter	Value	Default
Surface_name	string	–

Disable Opposed Normal Interaction Culling

Syntax

Disable Opposed Normal Interaction Culling

Summary

This line command disables the culling of interactions that do not have opposed normals.

Description

This is a temporary developer command

Opposed Normal Threshold Angle

Syntax

Opposed Normal Threshold Angle *{=}* *Angle*

Summary

Assign the angle, in degrees, for which surface normals are considered to be opposed.

Description

Prescribed angle must be greater or equal to 90 degrees.

Warning: This option is not recommended for general use but serves as a temporary patch that can be used for some problems.

Parameter	Value	Default
{=}	{= | are | is}	–
Angle	real	–

Output Rule

Syntax

Output Rule *{=}* *{more verbose | none | summary | verbose}*

Summary

Toggle amount of information output from the search.

NONE

Nothing will be printed to the screen.

SUMMARY

A summary of the calculation will be printed.

VERBOSE

A detailed report of the calculation is printed.

MORE VERBOSE

A more detailed report of the calculation is printed.

Parameter	Value	Default
{=}	{= | are | is}	–
OutputRule	{more verbose | none | summary | verbose}	–

Periodicity

Syntax

Periodicity For *surface {=}* Rectangular Offset = *offset1 offset2[offset3]*

Summary

Enables rectangular periodicity where specified surface is translated by {offset_x, offset_y, offset_z} before performing contact search.

Parameter	Value	Default
surface	"string"	–
{=}	{= | are | is}	–
offset	real1 real2[real3]	–

Periodicity

Syntax

Periodicity For *surface {=}* Align_Surface (Destination = *destination* \| [Invert_Normals = *invert_normals*])

Summary

Enables periodicity where specified surface is translated/rotated so that its centroid/normal align with the destination surface. By default the original face's normal is aligned to -1*destination normal, but this can be disabled by setting invert_normals to 0.

Parameter	Value	Default
surface	"string"	–
{=}	{= | are | is}	–
destination	"string"	–
invert_normals	integer	–

Periodicity

Syntax

Periodicity For *surface {=}* Cylindrical ([Axis = *axis1 axis2[axis3]*] \| [Center = *center1 center2[center3]*] \| Angle = *angle*)

Summary

Enables cylindrical periodicity where specified surface is rotated (angle in degrees) around the given axis before performing contact search.

Parameter	Value	Default
surface	"string"	–
{=}	{= | are | is}	–
axis	real1 real2[real3]	–
center	real1 real2[real3]	–
angle	real	–

Periodicity

Syntax

Periodicity For *surface* *{=}* Mapped ([X_Map = *x_map*] | [Y_Map = *y_map*] | [Z_Map = *z_map*])

Summary

Enables arbitrary periodicity where points p on specified surface are mapped via string functions p'=f(p) before performing contact search. Functions that are not specified default to a trivial mapping e.g. y->y

Parameter	Value	Default
surface	"string"	–
{=}	{= | are | is}	–
x_map	"string"	–
y_map	"string"	–
z_map	"string"	–

Restrict Contact To Subdomain Defined By Radius

Syntax

Restrict Contact To Subdomain Defined By Radius *r* And Point *VecX VecY VecZ*

Summary

Restrict contact to the subdomain defined by the radius and the point.

Parameter	Value	Default
r	real	–
VecX	real	–
VecY	real	–
VecZ	real	–

Skin All Blocks

Syntax

Skin All Blocks *{=}* *{off | on}* [Exclude *Blocks_to_exlude. . .*]

Summary

Informs the contact algorithms to skin all element blocks and then use the resulting surfaces in contact computations.

Parameter	Value	Default
{=}	{= | are | is}	–
Option	{off | on}	–

Update Search Every

Syntax

Update Search Every *n* Steps

Summary

Force an updating of the contact search every N steps.

Description

This command is useful if, for example, Aria is coupled to Adagio and the likelihood is strong that the mesh coordinates have changed significantly. Because contact surfaces may also appear in the definition of boundary conditions the application must selectively apply contact relations and boundary conditions appropriately. These issues are resolved automatically within the application code using the contact surface definitions and internal data masks.

Parameter	Value	Default
n	integer	–

Visualize Contact

Syntax

Visualize Contact

Summary

Output contact surfaces with facets marked with [0,1] corresponding to what percentage of their IPs where contact is applied.

Description

Enables output of surfaces where contact is active. Surface facets will be marked by the percentage of their IPs where contact constraints are applied - 0 meaning no IPs and 1 meaning all IPs.

Note: Visualization is enabled by default for contact defined using surface sidesets.

Warning: Use of this option is primarily for debugging purposes as the Results Output and History file will contain a skinned version of the contact element blocks. This may constitute a a large amount of information for discontiguously meshed models.

Warning: Using contact visualization with multiple contact definitions on the same surface can result in duplicating contact contributions. Additionally, toggling each contact definition independently can result in errors in contact visualization.

Interaction

Scope

Contact Definition

Summary

Contains the commands needed to define a surface-surface interaction.

```
begin Interaction Interaction_name

   Normal Tolerance {=} Distance

   Side A {=} Side a...

   Side B {=} Side b...

   Surfaces {=} Surfaces...

end Interaction Interaction_name
```

Line Commands

Normal Tolerance

Syntax

Normal Tolerance *{=} Distance*

Summary

Set distance of normal tolerance.

Parameter	Value	Default
{=}	{= | are | is}	–
Distance	real	–

Side A

Syntax

Side A *{=} Side a...*

Summary

Defines a list of surfaces that are side a to each surface in the side b list

Parameter	Value	Default
{=}	{= | are | is}	–
Side a	string. . .	–

Side B

Syntax

Side B *{=}* *Side b. . .*

Summary

Defines a list of surfaces that are side b to each surface in the side a list

Parameter	Value	Default
{=}	{= | are | is}	–
Side b	string. . .	–

Surfaces

Syntax

Surfaces *{=}* *Surfaces. . .*

Summary

Defines a pairwise set of surfaces for which to turn on interactions

Parameter	Value	Default
{=}	{= | are | is}	–
Surfaces	string. . .	–

Interaction Defaults

Scope

Contact Definition

Summary

Contains the interaction default commands

```
begin Interaction Defaults

   General Contact {=} {off | on}

   Self Contact {=} {off | on}

   Surfaces {=} Surfaces...

end Interaction Defaults
```

Line Commands

General Contact

Syntax

General Contact *{=} {off | on}*

Summary

Sets all contact that is not self contact to either what is defined in interaction behavior

Parameter	Value	Default
{=}	{= | are | is}	–
Option	{off | on}	–

Self Contact

Syntax

Self Contact *{=} {off | on}*

Summary

Specifies whether self contact is on or off

Parameter	Value	Default
{=}	{= | are | is}	–
Option	{off | on}	–

Surfaces

Syntax

Surfaces *{=} Surfaces. . .*

Summary

Defines a pairwise set of surfaces for which to turn on interactions

Parameter	Value	Default
{=}	{= | are | is}	–
Surfaces	string. . .	–

Search Options

Scope
Contact Definition

Summary
Contains the commands to set the search options

```
begin Search Options Search_options_name

   Normal Tolerance {=} Distance

end Search Options Search_options_name
```

Line Commands

Normal Tolerance

Syntax
Normal Tolerance *{=}* *Distance*

Summary
Set distance of normal tolerance.

Parameter	Value	Default
{=}	{= | are | is}	–
Distance	real	–

Enforcement

Scope
Contact Definition

Summary
Contains the commands needed to define contact enforcement for an analysis in Aria.

Description
This block command is used to define the type of contact between the surfaces. The name of the enforcement is specified by the user. When the contact includes some contact resistance, the conductance coefficient for the contact between the two surfaces must be specified.

```
begin Enforcement Enforcement_name
```

```
   Dash Penalty Factor {=} value

   Enforcement For EquationName {=} EnforcementModel [ ModelParams...  ]

   Integer Data Values...

   Real Data Values...

   Side B Variable b_field To a_field

   Slave Variable Slave_field To Master_field

   Use Toggle Block ToggleName [ {on} ElementBlockList...  ]

end Enforcement Enforcement_name
```

Line Commands

Dash Penalty Factor

Syntax

Dash Penalty Factor *{=}* *value*

Summary

Defines a penalty factor to use for the dash constraints

Parameter	Value	Default
{=}	{= | are | is}	–
value	real	–

Enforcement For

Syntax

Enforcement For *EquationName* *{=}* *EnforcementModel* [*ModelParams...*]

Summary

Defines the model for enforcement for the command block interactions, either tied or resistance/conductance contact. Tied contact can be of type generalized or Lagrange multiplier/MPC, where generalized contact is the preferred contact algorithm.

Description

This command line sets the type of contact problem for interactions defined in the current CONTACT DEFINITION command block and the equation to which the enforcement applies.

The Generalized contact enforcement model can be either GAP_CONDUCTANCE, CONDUCTANCE or TIED_DOF. In the case of resistance contact one must define a CONDUCTANCE COEFFICIENT model. To specify that TIED contact will be enforced for DOF my_DOF_name, the enforcement model will be TIED_my_DOF_name, where the solution DOF for EquationName is defined in the EQ command line.

Lagrange multiplier enforcement is specified simply as model TIED and must be accompanied by a SIDE B VARIABLE command line within the ENFORCEMENT command block.

DASH contact enforcement will be enforced only in a TIED sense.

Parameter	Value	Default
EquationName	string	–
{=}	{= | are | is}	–
EnforcementModel	string	–

Integer Data

Syntax

Integer Data *Values. . .*

Summary

List of integer data values to be used by the FORTRAN user subroutine. Copies of these values are provided to the subroutine hence changes to these values within the subroutine are not saved.

Parameter	Value	Default
Values	integer. . .	–

Real Data

Syntax

Real Data *Values. . .*

Summary

List of real data values to be used by the FORTRAN user subroutine. Copies of these values are provided to the subroutine hence changes to these values within the subroutine are not saved.

Parameter	Value	Default
Values	real. . .	–

Side B Variable

Syntax

Side B Variable *b_field* To *a_field*

Summary

Defines the variables involved in enforcement for this interaction.

Description

This line command sets maps side b to side a variables involved in the current tied contact definition.

Parameter	Value	Default
b_field	string	–
a_field	string	–

Slave Variable

Syntax

Slave Variable *Slave_field* To *Master_field*

Summary

Defines the variables involved in enforcement for this interaction.

Description

DEPRECATED COMMAND: This line command sets the master and slave variables involved in the current tied contact definition.

Parameter	Value	Default
Slave_field	string	–
Master_field	string	–

Use Toggle Block

Syntax

Use Toggle Block *ToggleName* [*{on} ElementBlockList. . .*]

Summary

Specification for toggling entities in the computational model based on Toggle Block parameters. When used at the region level, the list of element blocks to be toggled must be provided. Otherwise a listing of entities is not needed as the Toggle Block will be associated with the command line or the enclosing command block.

Parameter	Value	Default
ToggleName	string	–

Along with the lines listed above, enforcements can be specified in the *Enforcement* block with the syntax

```
Enforcement for [EQ] = [MODEL] [MODEL ARGS...]
```

See the list below for all possible **Enforcement** options.

Continuity

Tied_Ip_Continuity

Syntax

Enforcement For Continuity ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{=}* Tied_Ip_Continuity ()

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
{=}	{= | are | is}	–

Vector_String_Function

Syntax

Enforcement For Continuity ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{=}* Vector_String_Function (F_X = *f_x* | F_Y = *f_y* | [F_Z = *f_z*])

Summary

Value from a string function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
{=}	{= | are | is}	–
f_x	“string”	–
f_y	“string”	–
f_z	“string”	–

Current

Butler_Volmer

Syntax

Enforcement For Current ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{=}* Butler_Volmer (F = *f* | R = *r* | Alpha_A = *alpha_a* | Alpha_C = *alpha_c* | [Flux_Lim_A = *flux_lim_a*] | [Flux_Lim_C = *flux_lim_c*] | Sign = *sign* | [Use_Solid_Stress = *use_solid_stress*] | [Use_Liquid_Stress = *use_liquid_stress*] | [Trans = *trans*] | [Reaction_Name = *reaction_name*])

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
{=}	{= | are | is}	–
f	real	–
r	real	–
alpha_a	real	–
alpha_c	real	–
flux_lim_a	real	0
flux_lim_c	real	0
sign	integer	–
use_solid_stress	“string”	–
use_liquid_stress	“string”	–
trans	real	0.3
reaction_name	“string”	–

Butler_Volmer_Enforcement

Syntax

Enforcement For Current ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{=}* Butler_Volmer_Enforcement ([F = *f*] | [R = *r*] | Alpha_A = *alpha_a* | Alpha_C = *alpha_c* | [Flux_Lim_A = *flux_lim_a*] | [Flux_Lim_C = *flux_lim_c*] | [Toggle = *toggle*])

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
{=}	{= | are | is}	–
f	real	9.649e+04
r	real	8.314
alpha_a	real	–
alpha_c	real	–
flux_lim_a	real	0
flux_lim_c	real	0
toggle	"string"	–

Conductance

Syntax

Enforcement For Current ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{=}* Conductance [Toggle = *toggle*]

Summary

Applies a contact current flux of the form $C(V_1 - V_2)$ to sides 1 and 2 of the contact interface where C is a user specified conductance coefficient. Used in conjunction with a CONDUCTANCE COEFFICIENT model.

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
{=}	{= | are | is}	–
toggle	"string"	–

Gap_Conductance

Syntax

Enforcement For Current ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{=}* Gap_Conductance [Toggle = *toggle*]

Summary

Applies a contact current flux of the form $\alpha(\sigma_1 \nabla V_1 \cdot n_1 + \sigma_2 \nabla V_2 \cdot n_2 + \tilde{C}(V_1 - V_2))/2$ to sides 1 and 2 of the contact interface where $\alpha = 1 - \bar{C}/2C$. With C, a user defined coefficient and $\bar{C}$, an average conductance coefficient, then $\tilde{C} = 2C$ for $C \ll \bar{C}$ and $\bar{C}$ otherwise. Used in conjunction with a CONDUCTANCE COEFFICIENT model.

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
{=}	{= | are | is}	–
toggle	"string"	–

Nernst_Planck_Ion_Flux

Syntax

Enforcement For Current ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{=}* Nernst_Planck_Ion_Flux [Toggle = *toggle*]

Summary

Current enforcement for Nernst-Planck ion diffusion.

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
{=}	{= | are | is}	–
toggle	"string"	–

Scalar_String_Function

Syntax

Enforcement For Current ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{=}* Scalar_String_Function F = *f*

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
{=}	{= | are | is}	–
f	"string"	–

Tied_Voltage

Syntax

Enforcement For Current ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{=}* Tied_Voltage ([Co_Field = *co_field*] | [Toggle = *toggle*])

Summary

Applies a contact current flux of the form $(\sigma_1 \nabla V_1 \cdot n_1 + \sigma_2 \nabla V_2 \cdot n_2 + \bar{C}(V_1 - V_2))/2$ to sides 1 and 2 of the contact interface where σ is the electrical conductivity and $\bar{C}$ is an average conductance coefficient.

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
{=}	{= | are | is}	–
co_field	integer	0
toggle	"string"	–

Cvfem_Continuity

Tied_Continuity

Syntax

Enforcement For Cvfem_Continuity ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{=}* Tied_Continuity ()

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
{=}	{= | are | is}	–

Vector_String_Function

Syntax

Enforcement For Cvfem_Continuity ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{=}* Vector_String_Function (F_X = *f_x* | F_Y = *f_y* | [F_Z = *f_z*])

Summary

Value from a string function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
{=}	{= | are | is}	–
f_x	“string”	–
f_y	“string”	–
f_z	“string”	–

Cvfem_Energy

Gap_Conductance

Syntax

Enforcement For Cvfem_Energy ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{=}* Gap_Conductance [Toggle = *toggle*]

Summary

Applies a contact heat flux of the form $\alpha(k_1 \nabla T_1 \cdot n_1 + k_2 \nabla T_2 \cdot n_2 + \tilde{h}(T_1 - T_2))/2$ to sides 1 and 2 of the contact interface where $\alpha = 1 - \bar{h}/2h$. With h, a user defined coefficient and $\bar{h}$, an average conductance coefficient, then $\tilde{h} = 2h$ for $h \ll \bar{h}$ and $\bar{h}$ otherwise. Used in conjunction with a CONDUCTANCE COEFFICIENT model.

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
{=}	{= | are | is}	–
toggle	“string”	–

Scalar_String_Function

Syntax

Enforcement For Cvfem_Energy ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{=}* Scalar_String_Function F = *f*

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name.

Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
{=}	{= | are | is}	–
f	"string"	–

Tied_Temperature

Syntax

Enforcement For Cvfem_Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{=}* Tied_Temperature ([Co_Field = *co_field*] | [Toggle = *toggle*])

Summary

Applies a contact heat flux of the form $(k_1 \nabla T_1 \cdot n_2 + k_2 \nabla T_2 \cdot n_2 + \bar{h}(T_1 - T_2))/2$ to sides 1 and 2 of the contact interface where k is the thermal conductivity and $\bar{h}$ is an average conductance coefficient.

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
{=}	{= | are | is}	–
co_field	integer	0
toggle	"string"	–

Cvfem_Lumped_Projection

Tied_Gradp

Syntax

Enforcement For Cvfem_Lumped_Projection ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{=}* Tied_Gradp ()

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
{=}	{= | are | is}	–

Cvfem_Momentum

Tied_Momentum

Syntax

Enforcement For Cvfem_Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{=}* Tied_Momentum ()

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
{=}	{= | are | is}	–

Vector_String_Function

Syntax

Enforcement For Cvfem_Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{=}* Vector_String_Function (F_X = *f_x* | F_Y = *f_y* | [F_Z = *f_z*])

Summary

Value from a string function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
{=}	{= | are | is}	–
f_x	"string"	–
f_y	"string"	–
f_z	"string"	–

Cvfem_Projection

Tied_Gradp

Syntax

Enforcement For Cvfem_Projection ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{=}* Tied_Gradp ()

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
{=}	{= | are | is}	–

Cvfem_Specific_Dissipation_Rate

Scalar_String_Function

Syntax

Enforcement For Cvfem_Specific_Dissipation_Rate ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{=}* Scalar_String_Function F = *f*

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name.

Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
{=}	{= | are | is}	–
f	"string"	–

Tied_Sdr

Syntax

Enforcement For Cvfem_Specific_Dissipation_Rate ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{=}* Tied_Sdr ()

Summary

Tied_Sdr contact flux; CVFEM pressure stabilization included

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
{=}	{= | are | is}	–

Cvfem_Turbulent_Kinetic_Energy

Scalar_String_Function

Syntax

Enforcement For Cvfem_Turbulent_Kinetic_Energy ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{=}* Scalar_String_Function F = *f*

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name.

Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
{=}	{= | are | is}	–
f	"string"	–

Tied_Tke

Syntax

Enforcement For Cvfem_Turbulent_Kinetic_Energy ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{=}* Tied_Tke ()

Summary

Tied_Tke contact flux; CVFEM pressure stabilization included

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
{=}	{= | are | is}	–

Energy

Conductance

Syntax

Enforcement For Energy ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{=}* Conductance [Toggle = *toggle*]

Summary

Applies a contact heat flux of the form $C(T_1 - T_2)$ to sides 1 and 2 of the contact interface where C is a user specified conductance coefficient. Used in conjunction with a CONDUCTANCE COEFFICIENT model.

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
{=}	{= | are | is}	–
toggle	"string"	–

Conductance_Salgon

Syntax

Enforcement For Energy ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{=}* Conductance_Salgon (A = *a* | B = *b* | Xi = *xi* | [Toggle = *toggle*])

Summary

Applies a contact heat flux of the form $(2\hat{k}a)/(\pi b^2(1-1,14\xi))(T_1-T_2)$ to sides 1 and 2 of the contact interface where k is the thermal conductivity, $\hat{k} = (k_1+k_2)/(k_1k_2/)$ and ξ is a user specified parameter (Salgon,1997).

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
{=}	{= | are | is}	–
a	real	–
b	real	–
xi	real	–
toggle	"string"	–

Contact_Resistance

Syntax

Enforcement For Energy ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{=}* Contact_Resistance ([Opposite_Material_Phase = *opposite_material_phase*] | [Toggle = *toggle*])

Summary

Applies a contact heat flux of the form $h(T - T_{opp})$ where h is the gap conductance coefficient defined for the surface material and the material phase of T_{opp} is optionally specified.

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
{=}	{= | are | is}	–
opposite_material_phase	"string"	–
toggle	"string"	–

Contact_Resistance_From_Porous

Syntax

Enforcement For Energy ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{=}* Contact_Resistance_From_Porous Material_Phase = *material_phase*

Summary

Applies a contact heat flux of the form $h(T - T_{opp})$ to ONLY the side of the contact interface containing an ENERGY equation, where h is the gap conductance coefficient defined for the surface material and the material phase of T_{opp} is optionally specified. Used in conjunction with a CONTACT_RESISTANCE_FROM_ENERGY model on porous enthalpy.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
{=}	{= | are | is}	–
material_phase	“string”	–

Dash_Tied

Syntax

Enforcement For Energy ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{=}* Dash_Tied ()

Summary

Uses a penalty based method to tie the temperatures on either side of the contact interface. May only be used with the Dash search algorithm.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
{=}	{= | are | is}	–

Gap_Conductance

Syntax

Enforcement For Energy ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{=}* Gap_Conductance [Toggle = *toggle*]

Summary

Applies a contact heat flux of the form $\alpha(k_1 \nabla T_1 \cdot n_1 + k_2 \nabla T_2 \cdot n_2 + \tilde{h}(T_1 - T_2))/2$ to sides 1 and 2 of the contact interface where $\alpha = 1 - \bar{h}/2h$. With h, a user defined coefficient and $\bar{h}$, an average conductance coefficient, then $\tilde{h} = 2h$ for $h \ll \bar{h}$ and $\bar{h}$ otherwise. Used in conjunction with a CONDUCTANCE COEFFICIENT model.

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
{=}	{= | are | is}	–
toggle	“string”	–

Gap_Conductance_From_Porous

Syntax

Enforcement For Energy ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{=}* Gap_Conductance_From_Porous (Material_Phase = *material_phase* | [Toggle = *toggle*])

Summary

Applies a contact heat flux of the form $\alpha(k_1 \nabla T_1 \cdot n_1 + k_2 \nabla T_2 \cdot n_2 + \tilde{h}(T_1 - T_2))/2$ to ONLY the side of the contact interface containing an ENERGY equation, where $\alpha = 1 - \bar{h}/2h$. For the side containing a porous enthalpy eq, MATERIAL_PHASE specifies which phase to consider. With h, a user defined coefficient and $\bar{h}$, an average conductance coefficient, then $\tilde{h} = 2h$ for $h \ll \bar{h}$ and $\bar{h}$ otherwise. Used in conjunction with a CONDUCTANCE COEFFICIENT

model and with a GAP_CONDUCTANCE_FROM_ENERGY model on porous enthalpy.

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
{=}	{= | are | is}	–
material_phase	"string"	–
toggle	"string"	–

Nitche_Tied

Syntax

Enforcement For Energy ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{=}* Nitche_Tied ([Co_Field = *co_field*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Summary

Applies a contact heat flux as described in "A robust Nitsches formulation for interface problems" by Annavarapu et al. 2012.

Description

The numerical heat flux $\hat{\mathbf{q}}$ at the contact interface is defined as

$$\hat{\mathbf{q}} = \alpha_h \, [T] + \{\mathbf{q}\}_\gamma \, ,$$

where α_h is a mesh dependent stabilization parameter, $[T] = T_1\mathbf{n}_1 + T_2\mathbf{n}_2$ denotes the scalar jump operator, and $\{\mathbf{q}\}_\gamma$ is a weighted averaging operator across the interface defined as

$$\{\mathbf{q}\}_\gamma = \gamma_1\mathbf{q}_1 + \gamma_2\mathbf{q}_2,$$

where $\mathbf{q}_i$ is the heat flux, and $\gamma_1 + \gamma_2 = 1$ are weighting parameters that sum to

one defined as

$$\gamma_i = \frac{\frac{\Omega_i}{k_i}}{\frac{\Omega_1}{k_1} + \frac{\Omega_2}{k_2}},$$

where Ω_i is the local element volume, and k_i is the thermal conductivity.

The mesh dependent stabilization parameter α_h is

$$\alpha_h = \frac{c\Gamma_{ij}}{\frac{\Omega_1}{k_1} + \frac{\Omega_2}{k_2}},$$

where c is a user specified multiplier, and $\Gamma_{ij} = \Gamma_1 + \Gamma_2$ is a measure of the local contact interface taken here as the sum of local contact face areas. If a tensor conductivity is provided, then a projected thermal conductivity is used i.e.,

$$k_i = \mathbf{n}_i^T \mathbf{K}_i \mathbf{n}_i.$$

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
{=}	{= | are | is}	–
co_field	integer	0
multiplier	real	1
toggle	"string"	–

Scalar_String_Function

Syntax

Enforcement For Energy ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{=}* Scalar_String_Function F = *f*

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
{=}	{= | are | is}	–
f	"string"	–

Shell_Enclosure

Syntax

Enforcement For Energy ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{=}* Shell_Enclosure (Enclosure = *enclosure* | [Toggle = *toggle*])

Summary

Applies a contact heat flux copied from the radiative flux field populated by Chaparral during an enclosure radiation solve. The flux field is interpolated from the contact surface to the current surface. Temperature coupling with the current surface temperature is also implemented using the method specified in the enclosure definition. The only parameter required is the name of the enclosure definition to couple with.

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
{=}	{= | are | is}	–
enclosure	"string"	–
toggle	"string"	–

Tied_D_Style_Rad_Temperature

Syntax

Enforcement For Energy ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{=}* Tied_D_Style_Rad_Temperature ([Lambda_Scaling = *lambda_scaling*] | [Toggle = *toggle*])

Summary

Applies a contact heat flux of the form $\lambda\sigma\epsilon(T_1^4 - T_2^4)$ to sides 1 and 2 of the contact interface where λ is a scaling parameter, σ is the stefan boltzmann constant ϵ is the emissivity. This model is similar to the DASH_TIED model but for radiation rather than convection.

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
{=}	{= | are | is}	–
lambda_scaling	real	1
toggle	"string"	–

Tied_D_Style_Temperature

Syntax

Enforcement For Energy ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{=}* Tied_D_Style_Temperature ([Lambda_Scaling = *lambda_scaling*] | [Co_Field = *co_field*] | [Toggle = *toggle*])

Summary

Applies a contact heat flux of the form $\lambda \bar{h}(T_1 - T_2)$ to sides 1 and 2 of the contact interface where λ is a penalty parameter, $\bar{h}$ is an average conductance coefficient. This model is similar to the DASH_TIED model.

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
{=}	{= | are | is}	–
lambda_scaling	real	1
co_field	integer	-1
toggle	“string”	–

Tied_Ip_Temperature

Syntax

Enforcement For Energy ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{=}* Tied_Ip_Temperature ([Co_Field = *co_field*] | [Toggle = *toggle*])

Summary

Tied temperature contact model with an additional interior penalty based jump term included.

Description

The numerical heat flux $\hat{\mathbf{q}}$ at the contact interface is defined as

$$\hat{\mathbf{q}} = \alpha_h \, [T] + \{\mathbf{q}\} \, ,$$

where α_h is a mesh dependent stabilization parameter, $[T] = T_1\mathbf{n}_1 + T_2\mathbf{n}_2$ denotes the scalar jump operator, and $\{\mathbf{q}\} = 0.5\,(\mathbf{q}_1 + \mathbf{q}_2)$ is the average heat flux across the interface.

The mesh dependent stabilization parameter α_h is defined similarly to the tied temperature contact model, except an additional interior penalty jump term is included i.e.,

$$\alpha_h = 0.5\bar{h} + \{k\}\,,$$

where the first term is the average surface conductance, and the second term is the interior penalty based jump term with $\{k\} = 0.5\,(k_1 + k_2)$ the average thermal conductivity across the interface.

If a tensor conductivity is provided, then a projected thermal conductivity is used in lieu of the scalar conductivity i.e.,

$$k_i = \mathbf{n}_i^T \mathbf{K}_i \mathbf{n}_i.$$

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle
If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
{=}	{= | are | is}	–
co_field	integer	0
toggle	"string"	–

Tied_Temperature

Syntax
Enforcement For Energy ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{=}* Tied_Temperature ([Co_Field = *co_field*] | [Toggle = *toggle*])

Summary
Applies a contact heat flux of the form $(k_1\nabla T_1 \cdot n_1 + k_2\nabla T_2 \cdot n_2 + \bar{h}(T_1 - T_2))/2$ to sides 1 and 2 of the contact interface where k is the thermal conductivity and $\bar{h}$ is an average conductance coefficient.

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
{=}	{= | are | is}	–
co_field	integer	0
toggle	"string"	–

Tied_To_Porous

Syntax

Enforcement For Energy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{=}* Tied_To_Porous ([Co_Field = *co_field*] | Material_Phase = *material_phase* | [Toggle = *toggle*])

Summary

Applies a contact heat flux of the form $(k_1 \nabla T_1 \cdot n_1 + k_2 \nabla T_2 \cdot n_2 + \bar{h}(T_1 - T_2))/2$ to ONLY the side of the contact interface containing an ENERGY equation, where k is the thermal conductivity and $\bar{h}$ is an average conductance coefficient. Used in conjunction with a TIED_TO_ENERGY model on porous enthalpy.

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
{=}	{= | are | is}	–
co_field	integer	0
material_phase	"string"	–
toggle	"string"	–

Mass_Balance

Conserved_Mass

Syntax

Enforcement For Mass_Balance ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{=}* Conserved_Mass [Co_Field = *co_field*]

Summary

Continuous mass balance and capillary pressure, discontinuous nonwetting phase saturation.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
{=}	{= | are | is}	–
co_field	integer	0

Pressure_Conductance

Syntax

Enforcement For Mass_Balance ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{=}* Pressure_Conductance [Toggle = *toggle*]

Summary

Applies a contact flux of the form $C(p_1 - p_2)$ to sides 1 and 2 of the contact

interface where C is a user specified conductance coefficient. Used in conjunction with a CONDUCTANCE COEFFICIENT model.

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
{=}	{= | are | is}	–
toggle	"string"	–

Scalar_String_Function

Syntax

Enforcement For Mass_Balance ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{=}* Scalar_String_Function F = *f*

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
{=}	{= | are | is}	–
f	"string"	–

Tied_Pressure

Syntax

Enforcement For Mass_Balance ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{=}* Tied_Pressure [Co_Field = *co_field*]

Summary

Applies a contact mass flux via tied pressure formulation analogous to the tied_temperature energy contact enforcement models, wherein temperature is replaced with pressure and heat flux replaced with mass flux

Description

The numerical mass flux $\hat{\mathbf{m}}$ at the contact interface is defined as

$$\hat{\mathbf{m}} = \frac{D_h}{2}\,[P] + \{\mathbf{m}\}\,,$$

where $\{\mathbf{m}\} = 0.5\,(\rho_1\mathbf{u}_1 + \rho_2\mathbf{u}_2)$ is the average mass flux, $[P] = P_1\hat{\mathbf{n}}_1 + P_2\hat{\mathbf{n}}_2$, and D_h is the averaged darcy mass flux coefficient analogous to the contact heat transfer coefficient.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
{=}	{= | are | is}	–
co_field	integer	0

Mesh

Tied_Mesh_Displacements

Syntax

Enforcement For Mesh ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{=}* Tied_Mesh_Displacements ()

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
{=}	{= | are | is}	–

Tied_Normal_Displacements

Syntax

Enforcement For Mesh ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{=}* Tied_Normal_Displacements ()

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
{=}	{= | are | is}	–

Vector_String_Function

Syntax

Enforcement For Mesh ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{=}* Vector_String_Function (F_X = *f_x* | F_Y = *f_y* | [F_Z = *f_z*])

Summary

Value from a string function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
{=}	{= | are | is}	–
f_x	“string”	–
f_y	“string”	–
f_z	“string”	–

Momentum

Tied_Momentum

Syntax

Enforcement For Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{=}* Tied_Momentum ()

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
{=}	{= | are | is}	–

Vector_String_Function

Syntax

Enforcement For Momentum ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{=}* Vector_String_Function (F_X = *f_x* | F_Y = *f_y* | [F_Z = *f_z*])

Summary

Value from a string function

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
{=}	{= | are | is}	–
f_x	“string”	–
f_y	“string”	–
f_z	“string”	–

Porous_Enthalpy

Contact_Resistance

Syntax

Enforcement For Porous_Enthalpy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{=}* Contact_Resistance [Opposite_Material_Phase = *opposite_material_phase*]

Summary

Applies a contact heat flux of the form $h(T - T_{opp})$ to both sides of the contact interface, where h is the gap conductance coefficient defined for the surface material.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
{=}	{= | are | is}	–
opposite_material_phase	“string”	–

Contact_Resistance_From_Energy

Syntax

Enforcement For Porous_Enthalpy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{=}* Contact_Resistance_From_Energy ()

Summary

Applies a contact heat flux of the form $h(T - T_{opp})$ to ONLY the side of the contact interface containing a POROUS_ENTHALPY equation, where h is the gap conductance coefficient defined for the surface material. Used in conjunction with a CONTACT_RESISTANCE_FROM_POROUS model on energy.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
{=}	{= | are | is}	–

Gap_Conductance

Syntax

Enforcement For Porous_Enthalpy ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{=}* Gap_Conductance ([Opposite_Material_Phase = *opposite_material_phase*] | [Toggle = *toggle*])

Summary

Applies a contact heat flux of the form $\alpha(k_1 \nabla T_1 \cdot n_1 + k_2 \nabla T_2 \cdot n_2 + \tilde{h}(T_1 - T_2))/2$ to both sides of the contact interface, where $\alpha = 1 - \bar{h}/2h$. With h, a user defined coefficient and $\bar{h}$, an average conductance coefficient, then $\tilde{h} = 2h$ for $h \ll \bar{h}$ and $\bar{h}$ otherwise. Used in conjunction with a CONDUCTANCE COEFFICIENT model.

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
{=}	{= | are | is}	–
opposite_material_phase	“string”	–
toggle	“string”	–

Gap_Conductance_From_Energy

Syntax

Enforcement For Porous_Enthalpy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{=}* Gap_Conductance_From_Energy [Toggle = *toggle*]

Summary

Applies a contact heat flux of the form $\alpha(k_1 \nabla T_1 \cdot n_1 + k_2 \nabla T_2 \cdot n_2 + \tilde{h}(T_1 - T_2))/2$ to ONLY the side of the contact interface containing a POROUS_ENTHALPY equation, where $\alpha = 1 - \bar{h}/2h$. With h, a user defined coefficient and $\bar{h}$, an average conductance coefficient, then $\tilde{h} = 2h$ for $h \ll \bar{h}$ and $\bar{h}$ otherwise. Used in conjunction with a CONDUCTANCE COEFFICIENT model and with a GAP_CONDUCTANCE_FROM_POROUS model on energy.

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
{=}	{= | are | is}	–
toggle	“string”	–

Scalar_String_Function

Syntax

Enforcement For Porous_Enthalpy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{=}* Scalar_String_Function F = *f*

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name.

Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
{=}	{= | are | is}	–
f	"string"	–

Shell_Enclosure

Syntax

Enforcement For Porous_Enthalpy ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{=}* Shell_Enclosure (Enclosure = *enclosure* | [Toggle = *toggle*])

Summary

Applies a contact heat flux copied from the radiative flux field populated by Chaparral during an enclosure radiation solve. The flux field is interpolated from the contact surface to the current surface. Temperature coupling with the current surface temperature is also implemented using the method specified in the enclosure definition. The only parameter required is the name of the enclosure definition to couple with.

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
{=}	{= | are | is}	–
enclosure	"string"	–
toggle	"string"	–

Tied_Temperature

Syntax

Enforcement For Porous_Enthalpy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{=}* Tied_Temperature ([Co_Field = *co_field*] | [Opposite_Material_Phase = *opposite_material_phase*] | [Toggle = *toggle*])

Summary

Applies a contact heat flux of the form $(k_1 \nabla T_1 \cdot n_1 + k_2 \nabla T_2 \cdot n_2 + \bar{h}(T_1 - T_2))/2$ to both sides of the contact interface, where k is the thermal conductivity and $\bar{h}$ is an average conductance coefficient.

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
{=}	{= | are | is}	–
co_field	integer	0
opposite_material_phase	“string”	–
toggle	“string”	–

Tied_To_Energy

Syntax

Enforcement For Porous_Enthalpy ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{=}* Tied_To_Energy ([Co_Field = *co_field*] | [Toggle = *toggle*])

Summary

Applies a contact heat flux of the form $(k_1 \nabla T_1 \cdot n_1 + k_2 \nabla T_2 \cdot n_2 + \bar{h}(T_1 - T_2))/2$ to ONLY the side of the contact interface containing a POROUS_ENTHALPY equation, where k is the thermal conductivity and $\bar{h}$ is an average conductance coefficient. Used in conjunction with a TIED_TO_POROUS model on energy.

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
{=}	{= | are | is}	–
co_field	integer	0
toggle	"string"	–

Porous_Species

Butler_Volmer

Syntax

Enforcement For Porous_Species ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{=}* Butler_Volmer (F = *f* | R = *r* | Alpha_A = *alpha_a* | Alpha_C = *alpha_c* | [Flux_Lim_A = *flux_lim_a*] | [Flux_Lim_C = *flux_lim_c*] | Sign = *sign* | Stoich = *stoich* | [Use_Solid_Stress = *use_solid_stress*] | [Use_Liquid_Stress = *use_liquid_stress*] | [Trans = *trans*] | [Reaction_Name = *reaction_name*])

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
{=}	{= | are | is}	–
f	real	–
r	real	–
alpha_a	real	–
alpha_c	real	–
flux_lim_a	real	0
flux_lim_c	real	0
sign	integer	–
stoich	real	–
use_solid_stress	"string"	–
use_liquid_stress	"string"	–
trans	real	0.3
reaction_name	"string"	–

Darcy_Flow

Syntax

Enforcement For Porous_Species ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{=}* Darcy_Flow ()

Summary

Species flux enforcement for Darcy flow.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
{=}	{= | are | is}	–

Nernst_Planck

Syntax

Enforcement For Porous_Species ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{=}* Nernst_Planck ()

Summary

Species flux enforcement for Nernst-Planck diffusion.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
{=}	{= | are | is}	–

Scalar_String_Function

Syntax

Enforcement For Porous_Species ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{=}* Scalar_String_Function F = *f*

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
{=}	{= | are | is}	–
f	“string”	–

Species

Butler_Volmer

Syntax

Enforcement For Species ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{=}* Butler_Volmer (F = *f* | R = *r* | Alpha_A = *alpha_a* | Alpha_C = *alpha_c* | [Flux_Lim_A = *flux_lim_a*] | [Flux_Lim_C = *flux_lim_c*] | Sign = *sign* | Stoich = *stoich* | [Use_Solid_Stress = *use_solid_stress*] | [Use_Liquid_Stress = *use_liquid_stress*] | [Trans = *trans*] | [Reaction_Name = *reaction_name*])

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
{=}	{= | are | is}	–
f	real	–
r	real	–
alpha_a	real	–
alpha_c	real	–
flux_lim_a	real	0
flux_lim_c	real	0
sign	integer	–
stoich	real	–
use_solid_stress	“string”	–
use_liquid_stress	“string”	–
trans	real	0.3
reaction_name	“string”	–

Scalar_String_Function

Syntax

Enforcement For Species ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{=}* Scalar_String_Function F = *f*

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name.

Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
{=}	{= | are | is}	–
f	"string"	–

Tied

Syntax

Enforcement For Species ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{=}* Tied [Co_Field = *co_field*]

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
{=}	{= | are | is}	–
co_field	integer	0

Voltage

Scalar_String_Function

Syntax

Enforcement For Voltage ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{=}* Scalar_String_Function F = *f*

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name.

Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
{=}	{= | are | is}	–
f	"string"	–

Tied_Voltage

Syntax

Enforcement For Voltage ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{=}* Tied_Voltage ([Co_Field = *co_field*] | [Toggle = *toggle*])

Summary

Applies a contact electric displacement flux of the form $(\epsilon_1 \nabla V_1 \cdot n_1 + \epsilon_2 \nabla V_2 \cdot n_2 + \bar{C}(V_1 - V_2))/2$ to sides 1 and 2 of the contact interface where ϵ is the electrical permittivity and $\bar{C}$ is an average conductance coefficient.

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
{=}	{= | are | is}	–
co_field	integer	0
toggle	"string"	–

Similarly, contact material models can be specified in the *Enforcement* block with the syntax

```
[MATERIAL PROPERTY] = [MODEL] [MODEL ARGS...]
```

See the list below for all possible **Material Model** options.

Contact Gap Temperature

Average

Syntax

Contact Gap Temperature ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) = Average [Equations = *Equations...*]

Summary

Average temperature across the interface

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Equations	string...	–

Constant

Syntax

Contact Gap Temperature ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) = Constant (*{t | value}* = *t* | [Equations = *Equations...*])

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
t	real	–
Equations	string...	–

Copied

Syntax

Contact Gap Temperature ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) = Copied (Source = *source* | [Equations = *Equations. . .*])

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
source	"string"	–
Equations	string. . .	–

Exponential

Syntax

Contact Gap Temperature ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent* | [Equations = *Equations. . .*])

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–
Equations	string. . .	–

Global

Syntax

Contact Gap Temperature ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) = Global (Global_Name = *global_name* | [Component = *component*] | [Equations = *Equations. . .*])

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
global_name	"string"	–
component	integer	0
Equations	string. . .	–

Nd_Table

Syntax

Contact Gap Temperature ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*] | [Equations = *Equations. . .*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–
Equations	string. . .	–

Polynomial

Syntax

Contact Gap Temperature ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*] | [Equations = *Equations. . .*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0
Equations	string. . .	–

Scalar_String_Function

Syntax

Contact Gap Temperature ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) = Scalar_String_Function (F = *f* | [Equations = *Equations. . .*])

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
f	"string"	–
Equations	string. . .	–

User_Field

Syntax

Contact Gap Temperature ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*] | [Equations = *Equations. . .*])

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–
Equations	string. . .	–

User_Function

Syntax

Contact Gap Temperature ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*] | [Equations = *Equations. . .*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1
toggle	"string"	–
Equations	string. . .	–

User_Plugin

Syntax

Contact Gap Temperature ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) = User_Plugin (Name = *Name* | [*plugin_parameters*] | [Equations = *Equations. . .*])

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Name	"string"	–
plugin_parameters	"string"	–
Equations	string. . .	–

Contact Gap Voltage

Average

Syntax

Contact Gap Voltage ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) = Average [Equations = *Equations. . .*]

Summary

Average voltage across the interface

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Equations	string. . .	–

Constant

Syntax

Contact Gap Voltage ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) = Constant (*{v | value}* = *v* | [Equations = *Equations. . .*])

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
v	real	–
Equations	string. . .	–

Copied

Syntax

Contact Gap Voltage ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) = Copied (Source = *source* | [Equations = *Equations. . .*])

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
source	"string"	–
Equations	string. . .	–

Exponential

Syntax

Contact Gap Voltage ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent* | [Equations = *Equations. . .*])

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–
Equations	string. . .	–

Global

Syntax

Contact Gap Voltage ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) = Global (Global_Name = *global_name* | [Component = *component*] | [Equations = *Equations. . .*])

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
global_name	"string"	–
component	integer	0
Equations	string. . .	–

Nd_Table

Syntax

Contact Gap Voltage ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*] | [Equations = *Equations. . .*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–
Equations	string. . .	–

Polynomial

Syntax

Contact Gap Voltage ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*] | [Equations = *Equations. . .*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0
Equations	string. . .	–

Scalar_String_Function

Syntax

Contact Gap Voltage ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) = Scalar_String_Function (F = *f* | [Equations = *Equations. . .*])

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
f	"string"	–
Equations	string. . .	–

User_Field

Syntax

Contact Gap Voltage ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*] | [Equations = *Equations. . .*])

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–
Equations	string. . .	–

User_Function

Syntax

Contact Gap Voltage ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*] | [Equations = *Equations. . .*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1
toggle	"string"	–
Equations	string. . .	–

User_Plugin

Syntax

Contact Gap Voltage ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) = User_Plugin (Name = *Name* | [*plugin_parameters*] | [Equations = *Equations. . .*])

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Name	"string"	–
plugin_parameters	"string"	–
Equations	string. . .	–

Contact Surface Pressure

Average

Syntax

Contact Surface Pressure ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) = Average (Source = *source* | [Equations = *Equations. . .*])

Summary

Average pressure across the interface

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
source	"string"	–
Equations	string. . .	–

Constant

Syntax

Contact Surface Pressure ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) = Constant (*{p | value}* = *p* | [Equations = *Equations. . .*])

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
p	real	–
Equations	string. . .	–

Copied

Syntax

Contact Surface Pressure ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) = Copied (Source = *source* | [Equations = *Equations. . .*])

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
source	"string"	–
Equations	string. . .	–

Either_Or

Syntax

Contact Surface Pressure ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) = Either_Or (Source = *source* | [Equations = *Equations. . .*])

Summary

Select expression found on either side of contact interface.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
source	"string"	–
Equations	string. . .	–

Exponential

Syntax

Contact Surface Pressure ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent* | [Equations = *Equations. . .*])

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–
Equations	string. . .	–

Global

Syntax

Contact Surface Pressure ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) = Global (Global_Name = *global_name* | [Component = *component*] | [Equations = *Equations. . .*])

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
global_name	"string"	–
component	integer	0
Equations	string. . .	–

Nd_Table

Syntax

Contact Surface Pressure ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*] | [Equations = *Equations. . .*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–
Equations	string. . .	–

Polynomial

Syntax

Contact Surface Pressure ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*] | [Equations = *Equations. . .*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0
Equations	string. . .	–

Scalar_String_Function

Syntax

Contact Surface Pressure ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) = Scalar_String_Function (F = *f* | [Equations = *Equations. . .*])

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
f	"string"	–
Equations	string. . .	–

User_Field

Syntax

Contact Surface Pressure ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*] | [Equations = *Equations. . .*])

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–
Equations	string. . .	–

User_Function

Syntax

Contact Surface Pressure ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*] | [Equations = *Equations. . .*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1
toggle	"string"	–
Equations	string. . .	–

User_Plugin

Syntax

Contact Surface Pressure ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) = User_Plugin (Name = *Name* | [*plugin_parameters*] | [Equations = *Equations. . .*])

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Name	"string"	–
plugin_parameters	"string"	–
Equations	string. . .	–

Gap Conductance Coefficient

Calore_User_Sub

Syntax

Gap Conductance Coefficient ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) = Calore_User_Sub (Name = *name* | Type = *type* | [Multiplier =

multiplier] | [Material_Data_Block = *material_data_block*] | [Data = *data*] | [Scaling_Field = *scaling_field*] | [Equations = *Equations. . .*])

Summary

Calore user subroutine contact gap conductance

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
name	"string"	–
type	"string"	–
multiplier	real	1
material_data_block	"string"	–
data	"string"	–
scaling_field	"string"	–
Equations	string. . .	–

Constant

Syntax

Gap Conductance Coefficient ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) = Constant (*{h | value}* = *h* | [Equations = *Equations. . .*])

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
h	real	–
Equations	string. . .	–

Copied

Syntax

Gap Conductance Coefficient ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) = Copied (Source = *source* | [Equations = *Equations. . .*])

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
source	"string"	–
Equations	string. . .	–

Exponential

Syntax

Gap Conductance Coefficient ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent* | [Equations = *Equations. . .*])

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–
Equations	string. . .	–

Fortran

Syntax

Gap Conductance Coefficient ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) = Fortran ([Multiplier = *multiplier*] | Sub_Name = *sub_name* | [Real_Data = *real_data*] | [Int_Data = *int_data*] | [Resource_Name = *resource_name*] | [Data = *data*] | [Equations = *Equations. . .*])

Summary

Fortran contact gap conductance subroutine.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
multiplier	real	1
sub_name	“string”	–
real_data	“string”	–
int_data	“string”	–
resource_name	“string”	–
data	“string”	–
Equations	string. . .	–

Global

Syntax

Gap Conductance Coefficient ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) = Global (Global_Name = *global_name* | [Component = *component*] | [Equations = *Equations. . .*])

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
global_name	“string”	–
component	integer	0
Equations	string. . .	–

Nd_Table

Syntax

Gap Conductance Coefficient ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*] | [Equations = *Equations. . .*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–
Equations	string. . .	–

Polynomial

Syntax

Gap Conductance Coefficient ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*] | [Equations = *Equations. . .*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0
Equations	string. . .	–

Scalar_String_Function

Syntax

Gap Conductance Coefficient ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) = Scalar_String_Function (F = *f* | [Equations = *Equations. . .*])

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
f	"string"	–
Equations	string. . .	–

User_Field

Syntax

Gap Conductance Coefficient ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*] | [Equations = *Equations. . .*])

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–
Equations	string. . .	–

User_Function

Syntax

Gap Conductance Coefficient ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*] | [Equations = *Equations. . .*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1
toggle	"string"	–
Equations	string. . .	–

User_Plugin

Syntax

Gap Conductance Coefficient ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) = User_Plugin (Name = *Name* | [*plugin_parameters*] | [Equations = *Equations. . .*])

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Name	"string"	–
plugin_parameters	"string"	–
Equations	string. . .	–

Offset Darcy Potential

Constant

Syntax

Offset Darcy Potential ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) = Constant (Offset = *offset* | Surface = *surface* | [Equations = *Equations. . .*])

Summary

Offset DARCY_POTENTIAL by constant value on surfaces p' = p + offset.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
offset	real	–
surface	"string"	–
Equations	string. . .	–

Scalar_String_Function

Syntax

Offset Darcy Potential ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) = Scalar_String_Function (Offset = *offset* | Surface = *surface* | [Equations = *Equations. . .*])

Summary

Offset DARCY_POTENTIAL by string function on surfaces p' = p + offset.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
offset	"string"	–
surface	"string"	–
Equations	string. . .	–

Offset Mesh Displacements

Constant

Syntax

Offset Mesh Displacements ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) = Constant ([Offset = *offset1 offset2[offset3]*] | [Scale = *scale*] | Surface = *surface* | [Equations = *Equations. . .*])

Summary

Offset MESH_DISPLACEMENTS by constant value on surfaces d' = scale*d + <offset_x,offset_y,offset_z>. If a scale is provided, both the quantity and its flux are scaled accordingly.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
offset	real1 real2[real3]	0
scale	real	1
surface	"string"	–
Equations	string. . .	–

Reflected

Syntax

Offset Mesh Displacements ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) = Reflected ([Normal = *normal1 normal2[normal3]*] | [Scale = *scale*] | Surface = *surface* | [Equations = *Equations. . .*])

Summary

Reflects MESH_DISPLACEMENTS along plane given by normal. If a scale is provided (default is -1), both the quantity and its flux are scaled accordingly along normal.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
normal	real1 real2[real3]	0
scale	real	-1
surface	"string"	–
Equations	string. . .	–

Vector_String_Function

Syntax

Offset Mesh Displacements ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) = Vector_String_Function ([Offset_X = *offset_x*] | [Offset_Y = *offset_y*] | [Offset_Z = *offset_z*] | [Scale = *scale*] | Surface = *surface* | [Equations = *Equations. . .*])

Summary

Offset MESH_DISPLACEMENTS by string function on surfaces d' = scale*d + <offset_x,offset_y,offset_z>. If a scale is provided, both the quantity and its flux are scaled accordingly.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
offset_x	"string"	–
offset_y	"string"	–
offset_z	"string"	–
scale	real	1
surface	"string"	–
Equations	string. . .	–

Offset Pressure

Constant

Syntax

Offset Pressure ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) = Constant (Offset = *offset* | Surface = *surface* | [Equations = *Equations. . .*])

Summary

Offset PRESSURE by constant value on surfaces p' = p + offset.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
offset	real	–
surface	"string"	–
Equations	string. . .	–

Scalar_String_Function

Syntax

Offset Pressure ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) = Scalar_String_Function (Offset = *offset* | Surface = *surface* | [Equations = *Equations. . .*])

Summary

Offset PRESSURE by string function on surfaces p' = p + offset.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
offset	"string"	–
surface	"string"	–
Equations	string. . .	–

Offset Solid Displacements

Constant

Syntax

Offset Solid Displacements ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) = Constant ([Offset = *offset1 offset2[offset3]*] | [Scale = *scale*] | Surface = *surface* | [Equations = *Equations. . .*])

Summary

Offset SOLID_DISPLACEMENTS by constant value on surfaces d' = scale*d +

<offset_x,offset_y,offset_z>. If a scale is provided, both the quantity and its flux are scaled accordingly.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
offset	real1 real2[real3]	0
scale	real	1
surface	"string"	–
Equations	string. . .	–

Reflected

Syntax

Offset Solid Displacements ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) = Reflected ([Normal = *normal1 normal2[normal3]*] | [Scale = *scale*] | Surface = *surface* | [Equations = *Equations. . .*])

Summary

Reflects SOLID_DISPLACEMENTS along plane given by normal. If a scale is provided (default is -1), both the quantity and its flux are scaled accordingly along normal.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
normal	real1 real2[real3]	0
scale	real	-1
surface	"string"	–
Equations	string. . .	–

Vector_String_Function

Syntax

Offset Solid Displacements ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) = Vector_String_Function ([Offset_X = *offset_x*] | [Offset_Y = *offset_y*] | [Offset_Z = *offset_z*] | [Scale = *scale*] | Surface = *surface* | [Equations = *Equations. . .*])

Summary

Offset SOLID_DISPLACEMENTS by string function on surfaces d' = scale*d + <offset_x,offset_y,offset_z>. If a scale is provided, both the quantity and its flux are scaled accordingly.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
offset_x	"string"	–
offset_y	"string"	–
offset_z	"string"	–
scale	real	1
surface	"string"	–
Equations	string. . .	–

Offset Species

Constant

Syntax

Offset Species ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) = Constant (Offset = *offset* | Surface = *surface* | [Equations = *Equations. . .*])

Summary

Offset SPECIES by constant value on surfaces s' = s + offset.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
offset	real	–
surface	"string"	–
Equations	string. . .	–

Scalar_String_Function

Syntax

Offset Species ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) = Scalar_String_Function (Offset = *offset* | Surface = *surface* | [Equations = *Equations. . .*])

Summary

Offset SPECIES by string function on surfaces s' = s + offset.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
offset	"string"	–
surface	"string"	–
Equations	string. . .	–

Offset Temperature

Constant

Syntax

Offset Temperature ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) = Constant (Offset = *offset* | Surface = *surface* | [Equations = *Equations. . .*])

Summary

Offset TEMPERATURE by constant value on surfaces T' = T + offset.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
offset	real	–
surface	"string"	–
Equations	string. . .	–

Scalar_String_Function

Syntax

Offset Temperature ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) = Scalar_String_Function (Offset = *offset* | Surface = *surface* | [Equations = *Equations. . .*])

Summary

Offset TEMPERATURE by string function on surfaces T' = T + offset.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
offset	"string"	–
surface	"string"	–
Equations	string. . .	–

Offset Velocity

Constant

Syntax

Offset Velocity ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) = Constant ([Offset = *offset1 offset2[offset3]*] | [Scale = *scale*] | Surface = *surface* | [Equations = *Equations. . .*])

Summary

Offset VELOCITY by constant value on surfaces v' = scale*v +

<offset_x,offset_y,offset_z>. If a scale is provided, both the quantity and its flux are scaled accordingly.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
offset	real1 real2[real3]	0
scale	real	1
surface	"string"	–
Equations	string. . .	–

Reflected

Syntax

Offset Velocity ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) = Reflected ([Normal = *normal1 normal2[normal3]*] | [Scale = *scale*] | Surface = *surface* | [Equations = *Equations. . .*])

Summary

Reflects VELOCITY along plane given by normal. If a scale is provided (default is -1), both the quantity and its flux are scaled accordingly along normal.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
normal	real1 real2[real3]	0
scale	real	-1
surface	"string"	–
Equations	string. . .	–

Vector_String_Function

Syntax

Offset Velocity ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) = Vector_String_Function ([Offset_X = *offset_x*] | [Offset_Y = *offset_y*] | [Offset_Z = *offset_z*] | [Scale = *scale*] | Surface = *surface* | [Equations = *Equations. . .*])

Summary

Offset VELOCITY by string function on surfaces v' = scale*v + <offset_x,offset_y,offset_z>. If a scale is provided, both the quantity and its flux are scaled accordingly.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
offset_x	"string"	–
offset_y	"string"	–
offset_z	"string"	–
scale	real	1
surface	"string"	–
Equations	string. . .	–

Offset Voltage

Constant

Syntax

Offset Voltage ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) = Constant (Offset = *offset* | Surface = *surface* | [Equations = *Equations. . .*])

Summary

Offset VOLTAGE by constant value on surfaces V' = V + offset.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
offset	real	–
surface	"string"	–
Equations	string. . .	–

Scalar_String_Function

Syntax

Offset Voltage ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) = Scalar_String_Function (Offset = *offset* | Surface = *surface* | [Equations = *Equations. . .*])

Summary

Offset VOLTAGE by string function on surfaces V' = V + offset.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
offset	"string"	–
surface	"string"	–
Equations	string. . .	–

User Expression

Binary_State

Syntax

User Expression ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) = Binary_State (User_Tag = *user_tag* | Location = *location* | [Initial_State = *initial_state*] | [Activation_Function = *activation_function*] | [Deactivation_Function = *deactivation_function*] | [Equations = *Equations. . .*])

Summary

Binary State user expression

Description

Provides an expression that represents the active/inactive state with criteria for transition. Parameters are:

User_Tag - Name of expression and field to provide

Location - Either ELEMENT to create an element field/expression or NODE to create a nodal field/expression

Initial State - Either ON/ACTIVE or OFF/INACTIVE

Activation Function - String function criteria that will set field state to active

Deactivation Function - String function criteria that will set field state to inactive

Either an activation function, deactivation function, or both must be provided.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
user_tag	"string"	–
location	"string"	–
initial_state	"string"	–
activation_function	"string"	–
deactivation_function	"string"	–
Equations	string. . .	–

Constant

Syntax

User Expression ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) = Constant (User_Tag = *user_tag* | Value = *value* | [Equations = *Equations. . .*])

Summary

Constant user expression

Description

Provides an expression with the name given by the user_tag parameter using the provided constant value. The provided user tag should not include operator, subindex, phase, component, or material phase prefixes or suffixes. See *General Naming Convention* for more detail about those.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
user_tag	"string"	–
value	real	–
Equations	string. . .	–

Copied

Syntax

User Expression ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) = Copied (User_Tag = *user_tag* | Source = *source* | [Model = *model*] | [Source_Species = *source_species*] | [Source_Phase_Label = *source_phase_label*] | [Source_Material_Phase_Label = *source_material_phase_label*] | [Equations = *Equations. . .*])

Summary

Copied user expression

Description

Provides an expression that is a copy of an existing expression. If none of the optional "SOURCE_SPECIES", "SOURCE_PHASE_LABEL" and "SOURCE_MATERIAL_PHASE_LABEL" parameters are provided then the provided "SOURCE" parameter will be decoded to extract the species name, phase, and material phase. For example "SOLID_PHASE_POROSITY" would be decoded to correspond to the POROSITY in the solid phase. However, if any of the optional "SOURCE_*" parameters are provided then the "SOURCE" parameter is treated only as the expression name.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
user_tag	"string"	–
source	"string"	–
model	"string"	–
source_species	"string"	–
source_phase_label	"string"	–
source_material_phase_label	"string"	–
Equations	string. . .	–

Cross_Product

Syntax

User Expression ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) = Cross_Product (User_Tag = *user_tag* | U = *u* | V = *v* | [Equations = *Equations. . .*])

Summary

Cross product user expression

Description

Takes the cross product of two vectors (u and v) and produces a vector, X where $X = u \times v$

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
user_tag	"string"	–
u	"string"	–
v	"string"	–
Equations	string. . .	–

Dot_Product

Syntax

User Expression ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) = Dot_Product (User_Tag = *user_tag* | A = *a* | B = *b* | [Equations = *Equations. . .*])

Summary

Dot product user expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
user_tag	"string"	–
a	"string"	–
b	"string"	–
Equations	string. . .	–

Exponential

Syntax

User Expression ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) = Exponential (User_Tag = *user_tag* | Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent* | [Equations = *Equations. . .*])

Summary

Exponential user expression

Description

Provides an expression with the name given by the user_tag parameter using the provided exponential function. The provided user tag should not include operator, subindex, phase, component, or material phase prefixes or suffixes. See *General Naming Convention* for more detail about those. The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
user_tag	"string"	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–
Equations	string. . .	–

From_Scalar_Elem_Field

Syntax

User Expression ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) = From_Scalar_Elem_Field (User_Tag = *user_tag* | Name = *name* | [Offset = *offset*] | [Equations = *Equations. . .*])

Summary

User expression from a scalar element field

Description

Provides an expression from element rank, edge rank, or face rank scalar fields. Use the offset to index into one of (potentially) several scalars stored at a mesh entity.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
user_tag	"string"	–
name	"string"	–
offset	integer	0
Equations	string. . .	–

Global

Syntax

User Expression ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) = Global (User_Tag = *user_tag* | Global_Name = *global_name* | [Component = *component*] | [Equations = *Equations. . .*])

Summary

Global user expression

Description

Provides an expression with the name given by the user_tag parameter using the provided global variable. The provided user tag should not include operator, subindex, phase, component, or material phase prefixes or suffixes. See *General Naming Convention* for more detail about those.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
user_tag	"string"	–
global_name	"string"	–
component	integer	0
Equations	string. . .	–

Nd_Table

Syntax

User Expression ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) = Nd_Table (User_Tag = *user_tag* | Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*] | [Equations = *Equations. . .*])

Summary

User expression from a multi-dimensional table.

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
user_tag	"string"	–
name	"string"	–
vars	"string"1["string"2["string"3["string"4["string"5]]]]	–
i0	"string"	–
i1	"string"	–
i2	"string"	–
x	"string"	–
y	"string"	–
z	"string"	–
Equations	string. . .	–

Outer_Product

Syntax

User Expression ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) = Outer_Product (User_Tag = *user_tag* | U = *u* | V = *v* | [Equations = *Equations. . .*])

Summary

Cross product user expression

Description

Takes the outer product of two vectors (u and v) and produces a tensor, by multiplying each element of u by each element of v

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
user_tag	"string"	–
u	"string"	–
v	"string"	–
Equations	string. . .	–

Polynomial

Syntax

User Expression ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) = Polynomial (User_Tag = *user_tag* | Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*] | [Equations = *Equations. . .*])

Summary

Polynomial user expression

Description

Provides an expression with the name given by the user_tag parameter using the provided polynomial function. The provided user tag should not include operator, subindex, phase, component, or material phase prefixes or suffixes. See *General Naming Convention* for more detail about those. The polynomial value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
user_tag	"string"	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0
Equations	string. . .	–

Rotation_Tensor

Syntax

User Expression ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) = Rotation_Tensor (User_Tag = *user_tag* | Theta = *theta* | [Nx = *nx*] | [Ny = *ny*] | [Nz = *nz*] | [Equations = *Equations. . .*])

Summary

Rotation tensor user expression

Description

Calculates a rotation tensor around a defined axis [nx, ny, nz] by an angle expression 'theta'. In 2D the axis is always [0, 0, 1]

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
user_tag	"string"	–
theta	"string"	–
nx	real	0
ny	real	0
nz	real	0
Equations	string. . .	–

Scalar_String_Function

Syntax

User Expression ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) = Scalar_String_Function (User_Tag = *user_tag* | F = *f* | [Equations = *Equations. . .*])

Summary

String function user expression

Description

Provides an expression with the name given by the user_tag parameter using the provided string function. The provided user tag should not include operator, subindex, phase, component, or material phase prefixes or suffixes. See *General Naming Convention* for more detail about those.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
user_tag	"string"	–
f	"string"	–
Equations	string. . .	–

Scalar_Tensor_Product

Syntax

User Expression ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) = Scalar_Tensor_Product (User_Tag = *user_tag* | Tensor = *tensor* | [Scalars = *scalars*] | [Multiplier = *multiplier*] | [Equations = *Equations. . .*])

Summary

Scalar tensor product

Description

Multiplies a tensor by a list of provided scalars (variables) and a multiplier (constant)

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
user_tag	"string"	–
tensor	"string"	–
scalars	"string"	–
multiplier	real	1
Equations	string. . .	–

Scalar_Vector_Product

Syntax

User Expression ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) = Scalar_Vector_Product (User_Tag = *user_tag* | Vector = *vector* | [Scalars = *scalars*] | [Multiplier = *multiplier*] | [Equations = *Equations. . .*])

Summary

Scalar vector product

Description

Multiplies a vector by a list of provided scalars (variables) and a multiplier (constant)

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
user_tag	"string"	–
vector	"string"	–
scalars	"string"	–
multiplier	real	1
Equations	string. . .	–

Tensor_Determinant

Syntax

User Expression ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) = Tensor_Determinant (User_Tag = *user_tag* | Tensor = *tensor* | [Equations = *Equations. . .*])

Summary

Tensor determinant

Description

Computes the determinant of the provided tensor

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
user_tag	"string"	–
tensor	"string"	–
Equations	string. . .	–

Tensor_String_Function

Syntax

User Expression ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) = Tensor_String_Function (User_Tag = *user_tag* | F_Xx = *f_xx* | F_Yy = *f_yy* | [F_Xy = *f_xy*] | [F_Zz = *f_zz*] | [F_Xz = *f_xz*] | [F_Yz = *f_yz*] | [F_Yx = *f_yx*] | [F_Zx = *f_zx*] | [F_Zy = *f_zy*] | [Equations = *Equations. . .*])

Summary

Tensor string function user expression

Description

Provides an expression with the name given by the user_tag parameter using the provided string function. The provided user tag should not include operator, subindex, phase, component, or material phase prefixes or suffixes. See *General Naming Convention* for more detail about those.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
user_tag	"string"	–
f_xx	"string"	–
f_yy	"string"	–
f_xy	"string"	0.0
f_zz	"string"	–
f_xz	"string"	0.0
f_yz	"string"	0.0
f_yx	"string"	0.0
f_zx	"string"	0.0
f_zy	"string"	0.0
Equations	string. . .	–

Tensor_String_Function_Symmetric

Syntax

User Expression ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) = Tensor_String_Function_Symmetric (User_Tag = *user_tag* | F_Xx = *f_xx* | F_Yy = *f_yy* | [F_Xy = *f_xy*] | [F_Zz = *f_zz*] | [F_Xz = *f_xz*] | [F_Yz = *f_yz*] | [Equations = *Equations. . .*])

Summary

Symmetric tensor string function user expression

Description

Provides an expression with the name given by the user_tag parameter using the provided string function. The provided user tag should not include operator, subindex, phase, component, or material phase prefixes or suffixes. See *General Naming Convention* for more detail about those.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
user_tag	"string"	–
f_xx	"string"	–
f_yy	"string"	–
f_xy	"string"	0.0
f_zz	"string"	–
f_xz	"string"	0.0
f_yz	"string"	0.0
Equations	string. . .	–

Tensor_Sum

Syntax

User Expression ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) = Tensor_Sum (User_Tag = *user_tag* | Contributions = *contributions* | [Equations = *Equations. . .*])

Summary

Tensor sum

Description

Summation of a list of tensor and/or scalar contributions. Scalar sums are applied to all tensor components

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
user_tag	"string"	–
contributions	"string"	–
Equations	string. . .	–

Tensor_Tensor_Product

Syntax

User Expression ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) = Tensor_Tensor_Product (User_Tag = *user_tag* | A = *a* | B = *b* | [Equations = *Equations. . .*])

Summary

Calculates the product of two tensors

Description

Calculates the resulting tensor, C, from the tensor multiplication of A and B ($AB = C$)

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
user_tag	"string"	–
a	"string"	–
b	"string"	–
Equations	string. . .	–

Tensor_Vector_Product

Syntax

User Expression ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) = Tensor_Vector_Product (User_Tag = *user_tag* | Vector = *vector* | Tensor = *tensor* | [Scale = *scale*] | [Equations = *Equations. . .*])

Summary

Tensor vector product

Description

Calculates a vector resulting from a tensor-vector product with an optional scaling factor.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
user_tag	"string"	–
vector	"string"	–
tensor	"string"	–
scale	real	1
Equations	string. . .	–

Transform_Tensor

Syntax

User Expression ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) = Transform_Tensor (User_Tag = *user_tag* | U = *u* | R = *r* | [Equations = *Equations. . .*])

Summary

Cross product user expression

Description

Does a tensor transformation $U^{'} = R^{T}UR$ where R is an orthogonal rotation matrix and U is a tensor. If multiple rotation matrices are provided e.g. R ="Rz Ry Rx", then the rotations will be applied in order, e.g. $R = R_x R_y R_z$.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
user_tag	"string"	–
u	"string"	–
r	"string"	–
Equations	string. . .	–

User_Field

Syntax

User Expression ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) = User_Field (User_Tag = *user_tag* | Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*] | [Equations = *Equations. . .*])

Summary

User field user expression

Description

Provides an expression with the name given by the user_tag parameter using the provided user field. The provided user tag should not include operator, subindex, phase, component, or material phase prefixes or suffixes. See *General Naming Convention* for more detail about those.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
user_tag	"string"	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–
Equations	string. . .	–

User_Function

Syntax

User Expression ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) = User_Function (User_Tag = *user_tag* | Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*] | [Equations = *Equations. . .*])

Summary

User function user expression

Description

Provides an expression with the name given by the user_tag parameter using the provided user function. The provided user tag should not include operator,

subindex, phase, component, or material phase prefixes or suffixes. See *General Naming Convention* for more detail about those.

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
user_tag	"string"	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1
toggle	"string"	–
Equations	string. . .	–

User_Plugin

Syntax

User Expression ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) = User_Plugin (User_Tag = *user_tag* | Name = *Name* | [*plugin_parameters*] | [Equations = *Equations. . .*])

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
user_tag	"string"	–
Name	"string"	–
plugin_parameters	"string"	–
Equations	string. . .	–

Vector_String_Function

Syntax

User Expression ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) = Vector_String_Function (User_Tag = *user_tag* | F_X = *f_x* | F_Y = *f_y* | [F_Z = *f_z*] | [Equations = *Equations. . .*])

Summary

Vector string function user expression

Description

Provides an expression with the name given by the user_tag parameter using the provided string function. The provided user tag should not include operator, subindex, phase, component, or material phase prefixes or suffixes. See *General Naming Convention* for more detail about those.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
user_tag	"string"	–
f_x	"string"	–
f_y	"string"	–
f_z	"string"	–
Equations	string. . .	–

Vector_Sum

Syntax

User Expression ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) = Vector_Sum (User_Tag = *user_tag* | Contributions = *contributions* | [Equations = *Equations. . .*])

Summary

Vector sum

Description

Summation of a list of vector and/or scalar contributions. Scalar sums are applied to all vector components

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
user_tag	"string"	–
contributions	"string"	–
Equations	string...	–

7.24 Directed Energy

Directed Energy Model

Scope

Aria Region, Equation System, Explicit Equation System, Root Finder Equation System

Summary

Application of a directed energy model, which can be useful in simulating laser, electron beam, or plasma arc heating

```
begin Directed Energy Model Model Name

   Activation Temperature {=} value

   Add Surface value...

   Add Volume value...

   Deposition {=} {false | true}

   Integrated {=} {false | true}

   Origin And Normal Function {=} value

   Path Function {=} value

   Toggle {=} value

   begin Beer Lambert Source
   end
```

```
   begin Conical Source
   end

   begin Double Ellipsoidal Source
   end

   begin Ellipsoidal Source
   end

   begin Gaussian Power Weld Flux
   end

   begin Hemisphere Source
   end

   begin Parallelepiped Source
   end

   begin Sharp Power Weld Flux
   end

   begin Sphere Source
   end
```

end Directed Energy Model *Model Name*

Line Commands

Activation Temperature

Syntax

Activation Temperature *{=}* *value*

Summary

Specifies temperature in which an activation field for elements that have gone over this activation temperature will be maintained.

Parameter	Value	Default
{=}	{= | are | is}	–
value	real	0

Add Surface

Syntax

Add Surface *value. . .*

Summary

Specify the surfaces for the application of the directed energy model

Parameter	Value	Default
value	string. . .	–

Add Volume

Syntax

Add Volume *value. . .*

Summary

Specify the blocks for the application of the directed energy model

Parameter	Value	Default
value	string. . .	–

Deposition

Syntax

Deposition *{=} {false | true}*

Summary

Whether or not to disable source on elements that are not currently deposited. If true, the source will only be applied to elements with 'deposit' field active.

Parameter	Value	Default
{=}	{= | are | is}	–
value	{false | true}	–

Integrated

Syntax

Integrated *{=} {false | true}*

Summary

Whether or not to integrate the directed energy over the time-step to allow for larger time-steps. If true, the source will be integrated over the time step and the time average applied

Parameter	Value	Default
{=}	{= | are | is}	–
value	{false | true}	–

Origin And Normal Function

Syntax

Origin And Normal Function *{=}* *value*

Summary

Specifies the position and direction function as a function of time for a directed energy beam flux. Function column headings must be: time X Y Z nx ny nz where -(nx ny nz) defines the beam orientation.

Parameter	Value	Default
{=}	{= | are | is}	–
value	string	–

Path Function

Syntax

Path Function *{=}* *value*

Summary

Specifies a time and position function for which a laser source is applied in the named volume. Function column headings must be: time X Y Z nx ny nz where -(nx ny nz) defines the depth direction.

Parameter	Value	Default
{=}	{= | are | is}	–
value	string	–

Toggle

Syntax

Toggle *{=}* *value*

Summary

Specify the toggle block to use for the laser

Parameter	Value	Default
{=}	{= | are | is}	–
value	string	–

Beer Lambert Source

Summary

Selects the Beer Lambert laser source model. Refer to Section *Beer-Lambert* for more details.

```
begin Beer Lambert Source

   Beam Diameter {=} value

   Effective Beam Diameter {=} value

   Efficiency {=} value

end Beer Lambert Source
```

Line Commands

Beam Diameter

Syntax

Beam Diameter *{=} value*

Summary

Specifies laser spot size.

Parameter	Value	Default
{=}	{= | are | is}	–
value	real	0

Effective Beam Diameter

Syntax

Effective Beam Diameter *{=} value*

Summary

Specifies laser spot diameter.

Parameter	Value	Default
{=}	{= | are | is}	–
value	real	0

Efficiency

Syntax

Efficiency *{=} value*

Summary

Specifies fraction of power used in this source

Parameter	Value	Default
{=}	{= | are | is}	–
value	real	0

Conical Source

Summary

Selects the Conical laser source model. This model uses a Gaussian distribution type. Refer to *Conical* for more details.

```
begin Conical Source

   Beam Diameter {=} value

   Beam Diameter At Depth {=} value

   Efficiency {=} value

   Penetration Depth {=} value

end Conical Source
```

Line Commands

Beam Diameter

Syntax

Beam Diameter *{=}* *value*

Summary

Specifies laser spot size.

Parameter	Value	Default
{=}	{= | are | is}	–
value	real	0

Beam Diameter At Depth

Syntax

Beam Diameter At Depth *{=}* *value*

Summary

Specifies effective diameter at depth.

Parameter	Value	Default
{=}	{= | are | is}	–
value	real	0

Efficiency

Syntax

Efficiency *{=}* *value*

Summary

Specifies fraction of power used in this source

Parameter	Value	Default
{=}	{= | are | is}	–
value	real	0

Penetration Depth

Syntax

Penetration Depth *{=}* *value*

Summary

Specifies the depth of beam penetration.

Parameter	Value	Default
{=}	{= | are | is}	–
value	real	0

Double Ellipsoidal Source

Summary

Selects the Ellipsoid laser source model. This model uses a Gaussian distribution type. Refer to *Double Ellipsoid* for more details.

```
begin Double Ellipsoidal Source

   Aft Travel Semi Axis {=} value

   Depth Semi Axis {=} value

   Efficiency {=} value
```

```
   Fore Power Fraction {=} value

   Fore Travel Semi Axis {=} value

   Width Semi Axis {=} value

end Double Ellipsoidal Source
```

Line Commands

Aft Travel Semi Axis

Syntax

Aft Travel Semi Axis *{=}* *value*

Summary

Defines the aft (negative) travel direction semiaxis for the double ellipsoidal source model.

Parameter	Value	Default
{=}	{= | are | is}	–
value	real	0

Depth Semi Axis

Syntax

Depth Semi Axis *{=}* *value*

Summary

Defines the depth direction semiaxis for the ellipsoidal, double ellipsoidal and rectangular source model.

Parameter	Value	Default
{=}	{= | are | is}	–
value	real	0

Efficiency

Syntax

Efficiency *{=}* *value*

Summary

Specifies fraction of power used in this source

Parameter	Value	Default
{=}	{= | are | is}	–
value	real	0

Fore Power Fraction

Syntax

Fore Power Fraction *{=}* *value*

Summary

Defines the power fraction in the fore (positive travel direction) half ellipsoid in the double ellipsoidal source model.

Parameter	Value	Default
{=}	{= | are | is}	–
value	real	0

Fore Travel Semi Axis

Syntax

Fore Travel Semi Axis *{=}* *value*

Summary

Defines the fore (positive) travel direction semiaxis for the double ellipsoidal source model.

Parameter	Value	Default
{=}	{= | are | is}	–
value	real	0

Width Semi Axis

Syntax

Width Semi Axis *{=}* *value*

Summary

Defines the width direction semiaxis for the ellipsoidal, double ellipsoidal and rectangular source model.

Parameter	Value	Default
{=}	{= | are | is}	–
value	real	0

Ellipsoidal Source

Summary

Selects the Ellipsoid laser source model. This model uses a Gaussian distribution type. Refer to *Ellipsoid* for more details.

```
begin Ellipsoidal Source

   Depth Semi Axis {=} value

   Efficiency {=} value

   Travel Semi Axis {=} value

   Width Semi Axis {=} value

end Ellipsoidal Source
```

Line Commands

Depth Semi Axis

Syntax

Depth Semi Axis *{=}* *value*

Summary

Defines the depth direction semiaxis for the ellipsoidal, double ellipsoidal and rectangular source model.

Parameter	Value	Default
{=}	{= | are | is}	–
value	real	0

Efficiency

Syntax

Efficiency *{=}* *value*

Summary

Specifies fraction of power used in this source

Parameter	Value	Default
{=}	{= | are | is}	–
value	real	0

Travel Semi Axis

Syntax

Travel Semi Axis *{=}* *value*

Summary

Defines the travel direction semiaxis for the ellipsoidal, double ellipsoidal and rectangular source model.

Parameter	Value	Default
{=}	{= | are | is}	–
value	real	0

Width Semi Axis

Syntax

Width Semi Axis *{=}* *value*

Summary

Defines the width direction semiaxis for the ellipsoidal, double ellipsoidal and rectangular source model.

Parameter	Value	Default
{=}	{= | are | is}	–
value	real	0

Gaussian Power Weld Flux

```
begin Gaussian Power Weld Flux

   Beam Diameter {=} value

   Clip Diameter {=} value

   Effective Beam Diameter {=} value

   Efficiency {=} value

end Gaussian Power Weld Flux
```

Line Commands

Beam Diameter

Syntax

Beam Diameter *{=}* *value*

Summary

Specifies laser spot size.

Parameter	Value	Default
{=}	{= | are | is}	–
value	real	0

Clip Diameter

Syntax

Clip Diameter *{=}* *value*

Summary

Specifies laser spot size.

Parameter	Value	Default
{=}	{= | are | is}	–
value	real	0

Effective Beam Diameter

Syntax

Effective Beam Diameter *{=}* *value*

Summary

Specifies laser spot diameter.

Parameter	Value	Default
{=}	{= | are | is}	–
value	real	0

Efficiency

Syntax

Efficiency *{=}* *value*

Summary

Specifies fraction of power used in this source

Parameter	Value	Default
{=}	{= | are | is}	–
value	real	0

Hemisphere Source

Summary

Selects the HEMISPHERE laser source model. When used with a Gaussian DISTRIBUTION, all source types produce maximum intensity at the center of the laser spot. Aside from the ACTIVATION SPHERE model these sources are currently limited to paths aligned with the x,y,z coordinate directions. Refer to *Hemispherical* for more details

```
begin Hemisphere Source

   Beam Diameter {=} value

   Distribution {=} {gaussian | uniform}

   Efficiency {=} value

end Hemisphere Source
```

Line Commands

Beam Diameter

Syntax

Beam Diameter *{=} value*

Summary

Specifies laser spot size.

Parameter	Value	Default
{=}	{= | are | is}	–
value	real	0

Distribution

Syntax

Distribution *{=} {gaussian | uniform}*

Summary

Defines how the laser energy is deposited onto the source geometry. Generally

speaking the spatial distribution is GAUSSIAN centered on the source point but for fiber bundles one may opt for the top hat or UNIFORM distribution.

Parameter	Value	Default
{=}	{= | are | is}	–
value	{gaussian | uniform}	–

Efficiency

Syntax

Efficiency *{=}* *value*

Summary

Specifies fraction of power used in this source

Parameter	Value	Default
{=}	{= | are | is}	–
value	real	0

Parallelepiped Source

Summary

Selects the Parallelpiped laser source model. This model uses a uniform distribution type. Refer to *Parallelpiped* for more details.

```
begin Parallelepiped Source

   Efficiency {=} value

   Penetration Depth {=} value

   Travel Semi Axis {=} value

   Width Semi Axis {=} value

end Parallelepiped Source
```

Line Commands

Efficiency

Syntax

Efficiency *{=}* *value*

Summary

Specifies fraction of power used in this source

Parameter	Value	Default
{=}	{= | are | is}	–
value	real	0

Penetration Depth

Syntax

Penetration Depth *{=}* *value*

Summary

Specifies the depth of beam penetration.

Parameter	Value	Default
{=}	{= | are | is}	–
value	real	0

Travel Semi Axis

Syntax

Travel Semi Axis *{=}* *value*

Summary

Defines the travel direction semiaxis for the ellipsoidal, double ellipsoidal and rectangular source model.

Parameter	Value	Default
{=}	{= | are | is}	–
value	real	0

Width Semi Axis

Syntax

Width Semi Axis *{=}* *value*

Summary

Defines the width direction semiaxis for the ellipsoidal, double ellipsoidal and rectangular source model.

Parameter	Value	Default
{=}	{= | are | is}	–
value	real	0

Sharp Power Weld Flux

```
begin Sharp Power Weld Flux

   Beam Diameter {=} value

   Clip Diameter {=} value

   Efficiency {=} value

end Sharp Power Weld Flux
```

Line Commands

Beam Diameter

Syntax

Beam Diameter *{=}* *value*

Summary

Specifies laser spot size.

Parameter	Value	Default
{=}	{= | are | is}	–
value	real	0

Clip Diameter

Syntax

Clip Diameter *{=}* *value*

Summary

Specifies laser spot size.

Parameter	Value	Default
{=}	{= | are | is}	–
value	real	0

Efficiency

Syntax

Efficiency *{=}* *value*

Summary

Specifies fraction of power used in this source

Parameter	Value	Default
{=}	{= | are | is}	–
value	real	0

Sphere Source

Summary

Selects the Sphere laser source model. When used with a Gaussian DISTRIBUTION, all source types produce maximum intensity at the center of the laser spot. Aside from the ACTIVATION SPHERE model these sources are currently limited to paths aligned with the x,y,z coordinate directions. Refer to *Spherical* for more details

```
begin Sphere Source

   Beam Diameter {=} value

   Distribution {=} {gaussian | uniform}

   Efficiency {=} value

end Sphere Source
```

Line Commands

Beam Diameter

Syntax

Beam Diameter *{=}* *value*

Summary

Specifies laser spot size.

Parameter	Value	Default
{=}	{= | are | is}	–
value	real	0

Distribution

Syntax

Distribution *{=} {gaussian | uniform}*

Summary

Defines how the laser energy is deposited onto the source geometry. Generally speaking the spatial distribution is GAUSSIAN centered on the source point but for fiber bundles one may opt for the top hat or UNIFORM distribution.

Parameter	Value	Default
{=}	{= | are | is}	–
value	{gaussian | uniform}	–

Efficiency

Syntax

Efficiency *{=} value*

Summary

Specifies fraction of power used in this source

Parameter	Value	Default
{=}	{= | are | is}	–
value	real	0

Absorption and Power Commands

Along with the lines listed above, absorption and power can be specified in the *Directed Energy Model* block with the syntax

```
[PROPERTY] = [MODEL] [MODEL ARGS...]
```

See the list below for all possible options:

Absorption Coefficient

Constant

Syntax

Absorption Coefficient [*{of} SpeciesName*] = Constant Value = *value*

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
value	real	–

Copied

Syntax

Absorption Coefficient [*{of}* *SpeciesName*] = Copied Source = *source*

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	"string"	–

Exponential

Syntax

Absorption Coefficient [*{of}* *SpeciesName*] = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Absorption Coefficient [*{of} SpeciesName*] = Global (Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
global_name	“string”	–
component	integer	0

Nd_Table

Syntax

Absorption Coefficient [*{of} SpeciesName*] = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Absorption Coefficient [*{of}* *SpeciesName*] = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Absorption Coefficient [*{of}* *SpeciesName*] = Scalar_String_Function F = *f*

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	"string"	–

User_Field

Syntax

Absorption Coefficient [*{of}* *SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Absorption Coefficient [*{of}* *SpeciesName*] = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1
toggle	"string"	–

User_Plugin

Syntax

Absorption Coefficient [*{of} SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	"string"	–
plugin_parameters	"string"	–

Laser Absorption Coefficient

Constant

Syntax

Laser Absorption Coefficient [*{of} SpeciesName*] = Constant Value = *value*

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
value	real	–

Copied

Syntax

Laser Absorption Coefficient [*{of}* *SpeciesName*] = Copied Source = *source*

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	“string”	–

Depth_Enhanced

Syntax

Laser Absorption Coefficient [*{of}* *SpeciesName*] = Depth_Enhanced (File = *file* | Dir_X = *dir_x* | Dir_Y = *dir_y* | [Dir_Z = *dir_z*] | [Alpha = *alpha*] | R = *r*)

Summary

Specify an absorption coefficient that is weighted by a visibility field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
file	“string”	–
dir_x	real	–
dir_y	real	–
dir_z	real	0
alpha	real	1
r	real	–

Exponential

Syntax

Laser Absorption Coefficient [*{of}* *SpeciesName*] = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Laser Absorption Coefficient [*{of} SpeciesName*] = Global (Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Laser Absorption Coefficient [*{of} SpeciesName*] = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	“string”	–
vars	“string”1[“string”2[“string”3[“string”4[“string”5]]]]	–
i0	“string”	–
i1	“string”	–
i2	“string”	–
x	“string”	–
y	“string”	–
z	“string”	–

Polynomial

Syntax

Laser Absorption Coefficient [*{of}* *SpeciesName*] = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	“string”	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Laser Absorption Coefficient [*{of} SpeciesName*] = Scalar_String_Function F = *f*

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	"string"	–

User_Field

Syntax

Laser Absorption Coefficient [*{of} SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Laser Absorption Coefficient [*{of} SpeciesName*] = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1
toggle	"string"	–

User_Plugin

Syntax

Laser Absorption Coefficient [*{of} SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	"string"	–
plugin_parameters	"string"	–

Visibility_Field

Syntax

Laser Absorption Coefficient [*{of}* *SpeciesName*] = Visibility_Field (Field = *field* | Dir_X = *dir_x* | Dir_Y = *dir_y* | [Dir_Z = *dir_z*] | Alpha = *alpha*)

Summary

Specify an absorption coefficient that is weighted by a visibility field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
field	"string"	–
dir_x	real	–
dir_y	real	–
dir_z	real	0
alpha	real	–

Laser Power

Constant

Syntax

Laser Power [*{of}* *SpeciesName*] = Constant Value = *value*

Summary

Constant value

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
value	real	–

Copied

Syntax

Laser Power [*{of}* *SpeciesName*] = Copied Source = *source*

Summary

Copied value from another expression

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
source	"string"	–

Exponential

Syntax

Laser Power [*{of}* *SpeciesName*] = Exponential (Variable = *variable* | [Constant = *constant*] | [Multiplier = *multiplier*] | Exponent = *exponent*)

Summary

Value from an exponential function

Description

The value, ϕ, is calculated as $\phi = C + Me^{kX}$ where X is the variable, C is the constant, M is the multiplier, and k is the exponent. C and M are optional and default to 0 and 1, respectively, if omitted

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
constant	real	0
multiplier	real	1
exponent	real	–

Global

Syntax

Laser Power [*{of}* *SpeciesName*] = Global (Global_Name = *global_name* | [Component = *component*])

Summary

Value from a global variable

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
global_name	"string"	–
component	integer	0

Nd_Table

Syntax

Laser Power [*{of}* *SpeciesName*] = Nd_Table (Name = *name* | [Vars = *vars1[vars2[vars3[vars4[vars5]]]]*] | [I0 = *i0*] | [I1 = *i1*] | [I2 = *i2*] | [X = *x*] | [Y = *y*] | [Z = *z*])

Summary

Value from a multi-dimensional table

Description

See *Multi-dimensional Tables*.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
vars	"string"1["string"2["string"3["string"4["string"5]]]]	–
i0	"string"	–
i1	"string"	–
i2	"string"	–
x	"string"	–
y	"string"	–
z	"string"	–

Polynomial

Syntax

Laser Power [*{of}* *SpeciesName*] = Polynomial (Variable = *variable* | Order = *order* | [Variable_Offset = *variable_offset*] | [C0 = *c0*] | [C1 = *c1*] | [C2 = *c2*] | [C3 = *c3*] | [C4 = *c4*] | [C5 = *c5*] | [C6 = *c6*] | [C7 = *c7*] | [C8 = *c8*])

Summary

Value from a polynomial function

Description

The value, ϕ, is calculated as $\phi = \sum_i^n C_i(X + X_0)^i$ where X_0 is the variable offset, X is the variable, n is the order and the coefficients provided are C_i

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
variable	"string"	–
order	integer	–
variable_offset	real	0
c0	real	0
c1	real	0
c2	real	0
c3	real	0
c4	real	0
c5	real	0
c6	real	0
c7	real	0
c8	real	0

Scalar_String_Function

Syntax

Laser Power [*{of} SpeciesName*] = Scalar_String_Function F = *f*

Summary

Value from a string function

Description

The value, ϕ, is calculated from the user-defined string function, f which can be a function of time (T), coordinates (X, Y, Z), or any valid expression name. Check the String Functions chapter (*String Functions*) for more details about valid functions and variables that can be used.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
f	"string"	–

User_Field

Syntax

Laser Power [*{of} SpeciesName*] = User_Field (Name = *name* | [Scaling = *scaling*] | [Global_Var = *global_var*] | [Time_Function = *time_function*])

Summary

Value from a user field

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
scaling	real	1
global_var	"string"	–
time_function	"string"	–

User_Function

Syntax

Laser Power [*{of} SpeciesName*] = User_Function (Name = *name* | X = *x* | [X_Multiplier = *x_multiplier*] | [Multiplier = *multiplier*] | [Toggle = *toggle*])

Summary

Value from a user function

Description

The following options are present in most boundary conditions/sources. The meaning of each option is as follows:

Toggle

If given, specifies which *toggle block* controls whether this BC is active or not.

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
name	"string"	–
x	"string"	–
x_multiplier	real	1
multiplier	real	1
toggle	"string"	–

User_Plugin

Syntax

Laser Power [*{of} SpeciesName*] = User_Plugin (Name = *Name* | [*plugin_parameters*])

Summary

Value from a user plugin

Parameter	Value	Default
{of}	{of | species | subindex}	–
SpeciesName	string	–
Name	"string"	–
plugin_parameters	"string"	–

7.25 Element Death

Element Death

Scope

Aria Region

Summary

Allows the specification of an element death criterion to one or more element volumes. The group of element volumes plus a death criterion make up a "death instance".

```
begin Element Death Death Name

   Add Volume Block Name...

   Criterion {=} {avg nodal value of | death criterion type undefined |␣
↪element value of | max nodal value of | min nodal value of} Variable␣
↪Name {< | <= | = | > | >= | death criterion compare type undefined}␣
↪Threshold Value

end Element Death Death Name
```

Line Commands

Add Volume

Syntax

Add Volume *Block Name. . .*

Summary

Specifies the names identifying an element volume to which an element death criterion is to be applied. Line can be repeated as needed to add multiple volumes. A group of element volumes plus a death criterion make up a "death instance".

Parameter	Value	Default
Block Name	string. . .	–

Criterion

Syntax

Criterion *{=} {avg nodal value of | death criterion type undefined | element value of | max nodal value of | min nodal value of} Variable Name {< | <= | = | > | >= | death criterion compare type undefined} Threshold Value*

Summary

The death criterion for one or more element volumes. Currently designed to support the specification of a threshold for a scalar variable defined on either a node or element. For nodal variables, various operations such as Min., Max. and Avg. are provided to reduce the nodal values to a single element value. For an element variable no operations are given as there is only one value per element. A group of element volumes plus a death criterion make up a "death instance". Currently only one criterion is allowed per death instance.

Parameter	Value	Default
{=}	{= | are | is}	–
DeathCriterion-Type	{avg nodal value of | death criterion type undefined | element value of | max nodal value of | min nodal value of}	–
Variable Name	string	–
DeathCriterion-CompareType	{< | <= | = | > | >= | death criterion compare type undefined}	–
Threshold Value	real	–

Cdfem Death

Scope

Aria Region

Summary

Allows the specification of an element death criterion to one or more element volumes. The group of element volumes plus a death criterion make up a “death instance”.

```
begin Cdfem Death Death Name

   Add Volume Block Name...

   Criterion {=} Variable Name {< | > | cdfem inequality criterion␣
↪undefined} Threshold Value

end Cdfem Death Death Name
```

Line Commands

Add Volume

Syntax

Add Volume *Block Name...*

Summary

Specifies the names identifying an element volume to which an element death criterion is to be applied. Line can be repeated as needed to add multiple volumes. A group of element volumes plus a death criterion make up a “death instance”.

Parameter	Value	Default
Block Name	string...	–

Criterion

Syntax

Criterion *{=} Variable Name {< | > | cdfem inequality criterion undefined} Threshold Value*

Summary

The death criterion for one or more element volumes. A group of element volumes plus a death criterion make up a “death instance”. Currently only one criterion is allowed per death instance.

Parameter	Value	Default
{=}	{= | are | is}	–
Variable Name	string	–
CDFEMInequalityCriterion	{< | > | cdfem inequality criterion undefined}	–
Threshold Value	real	–

Transfer Element Death

Syntax

Transfer Element Death

Summary

Cause execution of element death to occur at beginning of a time step rather than at the end of the step.

7.26 Enclosure Radiation

Viewfactor Calculation

Scope

Aria Region, Equation System, Explicit Equation System, Root Finder Equation System

Summary

This block command specifies a radiation enclosure and is used to define a method for calculating view factors. The parameter for this block corresponds to an instance of a radiation enclosure mechanics.

```
begin Viewfactor Calculation Vf_calc

   Bsp Tree Max Depth {=} depth And Min List Length {=} l

   Compute Rule {=} {hemicube | pairwise | read}

   Enclosure Quality Error Tolerance {=} Tolerance

   Enclosure Quality Warning Tolerance {=} Tolerance

   Geometric Tolerance {=} n

   Hemicube Max Subdivides {=} n

   Hemicube Min Separation {=} n
```

```
   Hemicube Resolution {=} n

   Number Of Rotations {=} n

   Output Rule {=} {more verbose | none | summary | verbose}

   Pairwise Monte Carlo Sample Rule {=} {halton | jitter | random |␣
↪uniform}

   Pairwise Monte Carlo Tol1 {=} Real_value

   Pairwise Monte Carlo Tol2 {=} Real_value

   Pairwise Number Of Monte Carlo Samples {=} n

   Pairwise Number Of Visibility Samples {=} n

   Pairwise Visibility Sample Rule {=} {halton | jitter | random |␣
↪uniform}

   X-Y Plane Symmetry

   X-Z Plane Symmetry

   Y-Z Plane Symmetry

end Viewfactor Calculation Vf_calc
```

Line Commands

Bsp Tree Max Depth

Syntax

Bsp Tree Max Depth *{=}* *depth* And Min List Length *{=}* *l*

Summary

This line command sets the BSP tree parameters.

Description

This line command sets the parameters for the Binary Space Partitioning (BSP) tree. This data structure is used to accelerate the geometric operations associated with visibility to calculate the view factors. The maximum depth of the tree will set the maximum number of partitioning boxes created for a given enclosure. The minimum list length will set the desired number faces to be

contained in a box. The depth of the tree will be increased until the minimum list length is met of the maximum depth is reached. It is suggested that these parameters not be adjusted from their default values.

Parameter	Value	Default
{=}	{= | are | is}	–
depth	integer	12
{=}	{= | are | is}	–
l	integer	25

Compute Rule

Syntax

Compute Rule *{=} {hemicube | pairwise | read}*

Summary

This line command sets the method for computing the view factors for this enclosure.

Parameter	Value	Default
{=}	{= | are | is}	–
VFComputeRule	{hemicube | pairwise | read}	HEMICUBE

Enclosure Quality Error Tolerance

Syntax

Enclosure Quality Error Tolerance *{=} Tolerance*

Summary

A viewfactor quality tolerance before an error is thrown

Description

A tolerance on enclosure quality defined as the area-weighted Linf of the raw rowsum minus 1 for the enclosures. If an enclosure is detected with a rowsum error larger than this tolerance an error is thrown and a list of the offending enclosures is printed to the aria log file. The default for this parameter is 0.05

Parameter	Value	Default
{=}	{= | are | is}	–
Tolerance	real	–

Enclosure Quality Warning Tolerance

Syntax

Enclosure Quality Warning Tolerance *{=} Tolerance*

Summary

A viewfactor quality tolerance before warnings are issued

Description

A tolerance on enclosure quality defined as the area-weighted Linf of the raw rowsum minus 1 for the enclosures. If an enclosure is detected with a rowsum error larger than this tolerance a warning message is printed in the aria log file along with a list of the offending enclosures. The default for this parameter is 1e-3

Parameter	Value	Default
{=}	{= | are | is}	–
Tolerance	real	1.0e-3

Geometric Tolerance

Syntax

Geometric Tolerance *{=}* *n*

Summary

Set the geometric tolerance

Description

This line command sets the spatial geometry tolerance for use in all geometry-related comparisons. This value should correspond to the known minimum geometric size of the element faces used to define the enclosure.

Parameter	Value	Default
{=}	{= | are | is}	–
n	real	1.0e-6

Hemicube Max Subdivides

Syntax

Hemicube Max Subdivides *{=}* *n*

Summary

Set the upper limit of hemicube subdivides

Description

This line command sets the upper limit of the number of element-face subdivides that will occur when the proximity criteria limit is exceeded (see the HEMICUBE MIN SEPARATION line command).

Parameter	Value	Default
{=}	{= | are | is}	–
n	integer	2

Hemicube Min Separation

Syntax

Hemicube Min Separation *{=}* *n*

Summary

Set the hemicube minimum separation

Description

This line command sets a non-dimensional minimum-separation distance allowed between element faces for the HEMICUBE method before the face is subdivided. The accuracy of the hemicube method degrades rapidly if the element faces are in close proximity to each other. This tolerance is based on dividing the normal distance between the center of the element faces by the effective diameter of the element face.

Parameter	Value	Default
{=}	{= | are | is}	–
n	real	5

Hemicube Resolution

Syntax

Hemicube Resolution *{=}* *n*

Summary

Set the hemicube resolution

Description

This line command sets the number of uniform sub-patches into which the hemicube will be divided.

Parameter	Value	Default
{=}	{= | are | is}	–
n	integer	400

Number Of Rotations

Syntax

Number Of Rotations *{=}* *n*

Summary

For an enclosure boundary defined with 2D axisymmetric geometry ONLY. Set the number of incremental segment rotations to define a 3D enclosure surface. Rotations are swept over 360 degrees for COMPUTE RULE Hemicube and over 180 degrees for COMPUTE RULE Pairwise.

Parameter	Value	Default
{=}	{= | are | is}	–
n	integer	1

Output Rule

Syntax

Output Rule *{=} {more verbose | none | summary | verbose}*

Summary

Selects the amount of information printed describing the view factor calculation

Description

By default a summary of the input values are output. The MORE VERBOSE option will print both the summary information and progress of the view factor calculation in percent done. The percent done is useful in estimating how much runtime should be allotted for the calculation.

Parameter	Value	Default
{=}	{= | are | is}	–
OutputRule	{more verbose | none | summary | verbose}	SUMMARY

Pairwise Monte Carlo Sample Rule

Syntax

Pairwise Monte Carlo Sample Rule *{=} {halton | jitter | random | uniform}*

Description

This line command selects the method for the distribution of Monte Carlo integration sample points on the surface for view-factor quadrature.

Parameter	Value	Default
{=}	{= | are | is}	–
AMCSampleRule	{halton | jitter | random | uniform}	HALTON

Pairwise Monte Carlo Tol1

Syntax

Pairwise Monte Carlo Tol1 *{=} Real_value*

Summary

Set first of two convergence checks for Monte Carlo integration

Description

This line command sets one of the two convergence checks used by the Monte Carlo integration algorithm for the PAIRWISE method. If the standard deviation

of the view factor divided by the value of the view factor is less than this tolerance, the Monte Carlo integration will terminate i.e

$$\frac{\text{Std. Dev. of} F_{ij}}{F_{ij}} < tol_1$$

Parameter	Value	Default
{=}	{= | are | is}	–
Real_value	real	1.0e-5

Pairwise Monte Carlo Tol2

Syntax

Pairwise Monte Carlo Tol2 *{=}* *Real_value*

Summary

Set second of two convergence checks for Monte Carlo integration

Description

This line command sets one of the two convergence checks used by the Monte Carlo integration algorithm for the PAIRWISE method. If the standard deviation of the view factor is less than this tolerance, the Monte Carlo integration will terminate i.e

$$\text{Std. Dev. of} F_{ij} < tol_2$$

Parameter	Value	Default
{=}	{= | are | is}	–
Real_value	real	1.0e-5

Pairwise Number Of Monte Carlo Samples

Syntax

Pairwise Number Of Monte Carlo Samples *{=}* *n*

Summary

Set the pairwise number of Monte Carlo sample points

Description

This line command specifies the number of Monte Carlo integration sample points when Monte Carlo integration is activated during the PAIRWISE view-factor-calculation algorithm.

Parameter	Value	Default
{=}	{= | are | is}	–
n	integer	1000

Pairwise Number Of Visibility Samples

Syntax

Pairwise Number Of Visibility Samples *{=}* *n*

Summary

Set the pairwise number visibility sample points

Description

This line command will set the number of sample points to use for evaluating visibility between element faces for the PAIRWISE view-factor-calculation algorithm. Visibility is used to select the method for calculating the view factors. If the two element faces are completely visible to each other, the view factor is calculated using Gauss quadrature or an analytic method. If the element faces are partially visible, Monte Carlo integration is used.

Parameter	Value	Default
{=}	{= | are | is}	–
n	integer	25

Pairwise Visibility Sample Rule

Syntax

Pairwise Visibility Sample Rule *{=}* *{halton | jitter | random | uniform}*

Summary

Set the pairwise visibility sample rule

Description

This line command selects the method for distributing the visibility sample points on the surface.

Parameter	Value	Default
{=}	{= | are | is}	–
AVSampleRule	{halton | jitter | random | uniform}	UNIFORM

X-Y Plane Symmetry

Syntax

X-Y Plane Symmetry

Summary

Specifies symmetry about the X-Y plane.

X-Z Plane Symmetry

Syntax

X-Z Plane Symmetry

Summary

Specifies symmetry about the X-Z plane.

Y-Z Plane Symmetry

Syntax

Y-Z Plane Symmetry

Summary

Specifies symmetry about the Y-Z plane.

Viewfactor Smoothing

Scope

Aria Region, Equation System, Explicit Equation System, Root Finder Equation System

Summary

Defines a view factor smoothing scheme and its associated parameters.

Description

This block command is used to define a method for smoothing the view factors. The defined method will be used for view factor smoothing only when requested with the USE VIEWFACTOR SMOOTHING line command in the ENCLOSURE DEFINITION command block.

When VIEWFACTOR SMOOTHING is requested a two step sequence is initiated

1) Reciprocity enforcement using the specified RECIPROCITY RULE
2) Rowsum enforcement as per the specified METHOD

In both steps the viewfactor matrix will be modified as prescribed.

Note: In step 1 the appropriate modifications are always applied. However, if the rowsum enforcement criteria of step 2 are not satisfied the viewfactor matrix supplied to the enclosure radiation calculation will be the viewfactor matrix resulting from step 1 and -not- the raw viewfactor matrix.

```
begin Viewfactor Smoothing Vf_smooth

   Convergence Tolerance {=} Param1

   Maximum Iterations {=} n
```

```
   Method {=} {least-squares | none | simple}

   Output Rule {=} {more verbose | none | summary | verbose}

   Reciprocity Rule {=} {addition | average | none | subtraction}

   Weight Power {=} Param1

end Viewfactor Smoothing Vf_smooth
```

Line Commands

Convergence Tolerance

Syntax

Convergence Tolerance *{=}* *Param1*

Summary

Sets convergence tolerance.

Description

This line command sets the convergence tolerance for both the SIMPLE and LEAST-SQUARES smoothing algorithms based on the following:

1) SIMPLE: $\left|1 - \sum F_{ij}\right| < tol$ 2) LEAST-SQUARES: $\|r\| < tol$

where r is the residual of the Lagrange multiplier matrix problem.

Parameter	Value	Default
{=}	{= | are | is}	–
Param1	real	1.0e-8

Maximum Iterations

Syntax

Maximum Iterations *{=}* *n*

Summary

Set maximum iterations.

Description

This line command sets the maximum number of iterations that the SIMPLE smoothing algorithm will take.

Parameter	Value	Default
{=}	{= | are | is}	–
n	integer	500

Method

Syntax

Method *{=} {least-squares | none | simple}*

Summary

Defines view factor smoothing method.

Description

This line command defines the algorithm for smoothing the view factors. Smoothing is a term used to describe the process of enforcing the row-sum property of the view-factor matrix given by

$$\sum F_{ij} = 1.0$$

and the reciprocity property of view factors given by

$$A_i F_{ij} = A_j F_{ji}.$$

The HEMICUBE method guarantees the row-sum property, but not necessarily the reciprocity property. The PAIRWISE method will guarantee the reciprocity property (lower diagonal is calculated and the upper triangular is filled in using reciprocity), but not necessarily the row-sum property. If these two properties are not met by the view-factor matrix, energy conservation will not be achieved.

Parameter	Value	Default
{=}	{= | are | is}	–
VFSmoothMethod	{least-squares | none | simple}	LEAST-SQUARES

Output Rule

Syntax

Output Rule *{=} {more verbose | none | summary | verbose}*

Summary

Set the amount of output.

Description

This line command sets the amount of information reported to the output screen about the view-factor smoothing.

Parameter	Value	Default
{=}	{= | are | is}	–
OutputRule	{more verbose | none | summary | verbose}	SUMMARY

Reciprocity Rule

Syntax

Reciprocity Rule *{=} {addition | average | none | subtraction}*

Summary

Selects reciprocity enforcement rule.

Description

This line command determines the method for enforcing reciprocity,

$$A_i F_{ij} = A_j F_{ji}$$

during smoothing. This command is only required for the HEMICUBE algorithm. The PAIRWISE algorithm uses reciprocity to fill in the lower triangular part of the view-factor matrix. If both of the view factors, F_{ij} and $F ji$, are nonzero, these factors are adjusted by averaging $A_i F_{ij}$ and $A_j F_{ji}$.

Parameter	Value	Default
{=}	{= | are | is}	–
VFMatrixSymmRule	{addition | average | none | subtraction}	AVERAGE

Weight Power

Syntax

Weight Power *{=} Param1*

Summary

Sets weight power.

Description

This line command sets the power, p, used for the weights in the LEAST-SQUARES smoothing algorithm

$$w_{ij} = F^p{}_{ij}.$$

Parameter	Value	Default
{=}	{= | are | is}	–
Param1	real	2.0

Radiosity Solver

Scope

Aria Region, Equation System, Explicit Equation System, Root Finder Equation System

Summary

Defines a radiosity solver to be used by the enclosure definition.

Description

This block command contains methods associated with the solution of the radiosity system and the linearization method for applying the radiative heat flux to the thermal model. Note that the parameter for this block command will be used with the USE RADIOSITY SOLVER line command in the ENCLOSURE DEFINITION command block.

```
begin Radiosity Solver Boundary condition instance name

   Convergence Tolerance {=} Tolerance

   Coupling {=} {lagged | mason | smoothed}

   Initial Guess {=} {physics | previous}

   Maximum Iterations {=} m

   Output Rule {=} {more verbose | none | summary | verbose}

   Polynomial Preconditioner Order {=} m

   Solver {=} {chaparral bicgstab | chaparral cg | chaparral gmres |␣
↪coupled}

end Radiosity Solver Boundary condition instance name
```

Line Commands

Convergence Tolerance

Syntax

Convergence Tolerance *{=} Tolerance*

Summary

Sets convergence tolerance.

Description

This line command sets the convergence tolerance for the iterative solution of the radiosity system. The iterative solution will stop when the ratio of the current residual to the initial residual is less than the specified tolerance:

$$\frac{\|r\|}{\|r_0\|} < tol$$

Parameter	Value	Default
{=}	{= | are | is}	–
Tolerance	real	1.0e-6

Coupling

Syntax

Coupling *{=} {lagged | mason | smoothed}*

Summary

Specifies linearization method.

Description

This line command specifies the linearization method used to apply the radiative heat flux to the thermal model.

Parameter	Value	Default
{=}	{= | are | is}	–
RadCouplingRule	{lagged | mason | smoothed}	MASON

Initial Guess

Syntax

Initial Guess *{=} {physics | previous}*

Summary

Default uses the previous solution state as an initial guess for the current solution.

Parameter	Value	Default
{=}	{= | are | is}	–
InitialGuess	{physics | previous}	PREVIOUS

Maximum Iterations

Syntax

Maximum Iterations *{=} m*

Summary

Sets maximum number of iterations.

Description

This line command sets the maximum number of iterations allowed for the iterative solution of the radiosity equations. If the maximum number of iterations is exceeded, the calculation will fail and the thermal simulation will stop.

Parameter	Value	Default
{=}	{= | are | is}	–
m	integer	300

Output Rule

Syntax

Output Rule *{=} {more verbose | none | summary | verbose}*

Summary

Sets the information reporting level.

Description

This line command will set the amount of information reported to the output screen about the radiosity solution.

Parameter	Value	Default
{=}	{= | are | is}	–
OutputRule	{more verbose | none | summary | verbose}	NONE

Polynomial Preconditioner Order

Syntax

Polynomial Preconditioner Order *{=} m*

Summary

Sets preconditioner polynomial order.

Description

This line command sets the polynomial preconditioner order for the iterative solution of the radiosity equations. This option only applies to the CHAPARRAL GMRES iterative solver.

Parameter	Value	Default
{=}	{= | are | is}	–
m	integer	20

Solver

Syntax

Solver *{=} {chaparral bicgstab | chaparral cg | chaparral gmres | coupled}*

Summary

Chaparral solver selection for radiosity system.

Description

This line command selects the method for solving the linear system of radiosity

equations. The currently available methods are iterative linear solvers.

Parameter	Value	Default
{=}	{= | are | is}	–
Rad-SolveRule	{chaparral bicgstab | chaparral cg | chaparral gmres | coupled}	CHAPARRAL CG

Enclosure Definition

Scope

Aria Region, Equation System, Explicit Equation System, Root Finder Equation System

Summary

Specifies a radiation enclosure. Corresponds to an instance of radiation enclosure mechanics.

Description

This block command is used to define an enclosure for the thermal model. There may be more than one enclosure defined for any thermal model. An enclosure definition includes the geometric aspects (list of element faces forming the surface), material properties, and setting of the algorithms for calculating the view factors between the element faces in the enclosure and the radiosity solution. The name of the enclosure is specified by the user.

** NOTE ** Enclosure names cannot begin with any sequence of numeric symbols (i.e.~0 - 9). This will cause a model to fail with very little in the way of error reporting.

```
begin Enclosure Definition Boundary condition instance name

   Activate Mean Beam Model With Bulk Node Model

   Add Surface SurfaceList...

   Area Output VariableName

   Blocking Surfaces

   Dash Closure Metric Samples {=} NumSamp [ Tol {=} Value  ]

   Dash Solve Enclosures {=} Values...

   Dash Subfaceting Algorithm {=} {dash_dicing | symmetric_imprinting}
```

```
  Database Name {=} Filename In {ascii | binary | pnetcdf} Format

  Disable Parallel Redistribution

  Emissivity {=} Value [ On SurfaceName  ]

  Emissivity Function {=} FunctionName [ On SurfaceName  ]

  Emissivity Subroutine {=} MySub [ On SurfaceName  ]

  Emissivity Time Function {=} FunctionName [ On SurfaceName  ]

  Enable Parallel Redistribution

  Input Database Name {=} Filename

  Integrated Flux Output VariableName

  Integrated Power Output VariableName

  Matched Flux On SurfaceName {=} FluxType [ Phase {=} MaterialPhaseName␣
↪ ]

  Mean Beam Length {=} l [Parameters]...

  Meshed Enclosure Is BlockList...

  Nonblocking Surfaces

  Output Database Name {=} Filename In {ascii | binary | pnetcdf} Format

  Overlapping Enclosure {=} {false | no | off | on | true | yes}

  Partial Enclosure Area {=} a

  Partial Enclosure Area Subroutine {=} FName

  Partial Enclosure Area Time Function {=} FName

  Partial Enclosure Emissivity {=} e

  Partial Enclosure Emissivity Subroutine {=} FName

  Partial Enclosure Emissivity Time Function {=} FName
```

```
   Partial Enclosure Flux Output VariableName

   Partial Enclosure Irradiance Output VariableName

   Partial Enclosure Radiosity Output VariableName

   Partial Enclosure Temperature {=} t

   Partial Enclosure Temperature Subroutine {=} FName

   Partial Enclosure Temperature Time Function {=} FName

   Preprocess Enclosures [ Options...  ]

   Radiosity Database Name {=} Filename

   Rowsum Database Name {=} Filename

   Topology Database Name {=} Filename

   Use Banded Wavelength Model ModelName

   Use Dash Enclosures [ type tol {=} Params...  ]

   Use Radiosity Solver Param0

   Use Skinned Shells On Surface SkinnedSurfaceList...

   Use Toggle Block ToggleName [ {on} ElementBlockList...  ]

   Use Viewfactor Calculation Param0

   Use Viewfactor Smoothing Param0

   Viewfactor Update {=} {affected_area | continuous | freeze | interval␣
↪| only_if_geometry_changed | standard | times} [ Using Params...  ]

   Viewfactor Update Start Time Is Start_time

   begin Band BandName
   end

end Enclosure Definition Boundary condition instance name
```

Line Commands

Activate Mean Beam Model With Bulk Node

Syntax

Activate Mean Beam Model With Bulk Node *Model*

Summary

Sets the bulk element representing the PMR material for the MBL (mean beam length) enclosure.

Parameter	Value	Default
Model	string	–

Add Surface

Syntax

Add Surface *SurfaceList. . .*

Summary

Adds surfaces, by name, to a boundary condition's extent.

Description

This line command is used to add surfaces to the extent of a boundary condition. In Exodus II, surfaces are specified as side sets, that have a global integer identifier. For example, side set 12 would be added by this line command using the surface name surface_12. Note that in SIERRA, each element of an array of strings must be separated by whitespace.

Parameter	Value	Default
SurfaceList	string. . .	–

Area Output

Syntax

Area Output *VariableName*

Summary

Output the area associated with this flux boundary condition.

Description

This line triggers output of the enclosure area and volume.

Parameter	Value	Default
VariableName	string	–

Blocking Surfaces

Syntax

Blocking Surfaces

Summary

Specifies a blocking enclosure

Description

This is the default behavior if neither this line command, nor the NONBLOCKING SURFACES line command are present. This line command informs Chaparral to consider blocking surfaces during the calculation of view factors. A blocking surface is defined as one or more element faces that occlude the view from any pair of element faces within the enclosure. If it is known that no blocking surfaces exist in the enclosure, the NONBLOCKING SURFACES line command can be included in the enclosure definition to reduce the compute time for view factors. Note that the BLOCKING SURFACES line command should be included if there is any doubt whether there are blocking surfaces in the enclosure.

Dash Closure Metric Samples

Syntax

Dash Closure Metric Samples *{=} NumSamp* [*Tol {=} Value*]

Summary

Specify number of samples for DASH closure metric and possibly an acceptable tolerance.

Description

When using DASH enclosures one may wish to assign the number of sample points used in determining closure. Optionally, one can set the acceptable value of closure metric associated with closure.

Parameter	Value	Default
{=}	{= | are | is}	–
NumSamp	integer	–

Dash Solve Enclosures

Syntax

Dash Solve Enclosures *{=} Values. . .*

Summary

List of integer index values for Dash enclosures to perform solution. The index values must be obtained by a preprocessing step using both PREPROCESS ENCLOSURE TOPOLOGY command line at the Region level and the PREPROCESS ENCLOSURES command line in the Enclosure Definition command block.

Parameter	Value	Default
{=}	{= | are | is}	–
Values	integer. . .	–

Dash Subfaceting Algorithm

Syntax

Dash Subfaceting Algorithm *{=} {dash_dicing | symmetric_imprinting}*

Summary

Specify the subfaceting algorithm type. This is a developer command.

Parameter	Value	Default
{=}	{= | are | is}	–
Type	{dash_dicing | symmetric_imprinting}	–

Database Name

Syntax

Database Name *{=} Filename* In *{ascii | binary | pnetcdf}* Format

Summary

Specifies common filename to read/write viewfactors

Description

This line command specifies the database name for both input and output of the view factors for this enclosure. The file name is provided by the user, and one of three file formats is specified.

The ASCII format should be used only for small enclosures. This format is useful when the analyst wants to examine the values of the view factors that have been evaluated.

The BINARY format will produce a machine-dependent file. This view-factor file can only be read on the same machine in subsequent calculations.

Both the ASCII and BINARY file formats may only be read on the same number of processors as they were written with, and are deprecated as of Aria version 4.43.5. The replacement PNETCDF format produces a single machine-independent file that may be written or read in runs using any number of processors.

Parameter	Value	Default
{=}	{= | are | is}	–
Filename	string	–
VFfileFormat	{ascii | binary | pnetcdf}	–

Disable Parallel Redistribution

Syntax

Disable Parallel Redistribution

Summary

Disable parallel redistribution of viewfactor matrix.

Description

Ensures that the default distribution of the viewfactor matrix is distribution of rows across MPI ranks is enforced - no parallel distribution of the view factor matrix.

Emissivity

Syntax

Emissivity *{=}* *Value* [On *SurfaceName*]

Summary

Sets a constant value of emissivity for a defined surface. If the optional parameters are not included, then this is the default emissivity for the enclosure. Otherwise it is only applied to the indicated surface.

Note: If a surface is called out more than once, the emissivity definition is overwritten: last one in wins.

Parameter	Value	Default
{=}	{= | are | is}	–
Value	real	–

Emissivity Function

Syntax

Emissivity Function *{=}* *FunctionName* [On *SurfaceName*]

Summary

Sets a emissivity function for a defined surface. If the optional parameters are not included, then this is the default emissivity for the enclosure. Otherwise it is only applied to the indicated surface.

Parameter	Value	Default
{=}	{= | are | is}	–
FunctionName	string	–

Emissivity Subroutine

Syntax

Emissivity Subroutine *{=} MySub* [On *SurfaceName*]

Summary

Sets a emissivity user subroutine for a defined surface. If the optional parameters are not included, then this is the default emissivity for the enclosure. Otherwise it is only applied to the indicated surface.

Also, the software supports using locally scoped user data for most user subroutines, but I haven't figured out a syntax for it here yet. So it is not yet supported. If you need to get data into this subroutine, use the region's "REAL DATA" and "INTEGER DATA" line commands.

Note: If a surface is called out more than once, the emissivity definition is overwritten: last one in wins.

Parameter	Value	Default
{=}	{= | are | is}	–
MySub	string	–

Emissivity Time Function

Syntax

Emissivity Time Function *{=} FunctionName* [On *SurfaceName*]

Summary

Sets a emissivity function of time for a defined surface. If the optional parameters are not included, then this is the default emissivity for the enclosure. Otherwise it is only applied to the indicated surface.

Parameter	Value	Default
{=}	{= | are | is}	–
FunctionName	string	–

Enable Parallel Redistribution

Syntax

Enable Parallel Redistribution

Summary

Enables parallel redistribution of viewfactor matrix.

Description

Invokes redistribution of the view factor matrix from a distribution of matrix rows across MPI ranks to a 2D distribution of row and column blocks to achieve

better performance in parallel. A current limitation of this option is that the Tpetra redistribution capability will use more memory than necessary and may not be a appropriate if one is memory constrained on a platform.

Input Database Name

Syntax

Input Database Name *{=}* *Filename*

Summary

Specifies filename to read viewfactors from

Description

This line command provides the name of the view-factor file. If this line command is present, the file containing the given view-factor matrix will be opened and read. The calculation of view factors will not take place. Note that you must also specify a value of READ for the COMPUTE RULE command in the associated VIEWFACTOR CALCULATION command block.

Parameter	Value	Default
{=}	{= | are | is}	–
Filename	string	–

Integrated Flux Output

Syntax

Integrated Flux Output *VariableName*

Summary

Calculate the average flux associated with this flux boundary condition.

Description

This line command specifies that, as a postprocess, the normal flux associated with this boundary condition be integrated over the surface to obtain the total power. This power is then divided by the total area of the surface to obtain the average flux on the surface, and stored in a global variable named "VariableName". This global variable may then be output to history files, or accessed in user subroutines, etc.

Parameter	Value	Default
VariableName	string	–

Integrated Power Output

Syntax

Integrated Power Output *VariableName*

Summary

Calculate the total power associated with this flux boundary condition.

Description

This line command specifies that, as a postprocess, the normal flux associated with this boundary condition be integrated over the surface to obtain the total power which is then stored into a global variable named “VariableName”. This global variable may then be output to history files, or accessed in user subroutines, etc.

Parameter	Value	Default
VariableName	string	–

Matched Flux On

Syntax

Matched Flux On *SurfaceName {=} FluxType* [Phase *{=} MaterialPhaseName*]

Summary

Indicates an equivalent matching flux for surfaces within the enclosure that border element blocks which do not have the ENERGY equation defined but do have an equivalent equation (such as ENTHALPY) defined. The primary use for this is in OMD simulations and should be considered at the same reliability level as OMD.

Parameter	Value	Default
SurfaceName	string	–
{=}	{= | are | is}	–
FluxType	string	–

Mean Beam Length

Syntax

Mean Beam Length *{=} l [Parameters]. . .*

Summary

Sets the expression for the radiant decay length L for the MBL (mean beam length) enclosure.

Parameter	Value	Default
{=}	{= | are | is}	–
l	string	–
Parameters	[string]. . .	–

Meshed Enclosure

Syntax

Meshed Enclosure Is *BlockList. . .*

Summary

Adds blocks, by name, to the meshed enclosure extent.

Description

In the event one wishes to solve a conduction problem within an enclosure as well as the radiation problem the interior discretization must be defined. This line command is used to add blocks to the extent of a enclosure interior. In Exodus II, surfaces are specified as side sets, that have a global integer identifier. For example, block 12 would be added by this line command using the surface name block_12. Note that in SIERRA, each element of an array of strings must be separated by whitespace.

Parameter	Value	Default
BlockList	string. . .	–

Nonblocking Surfaces

Syntax

Nonblocking Surfaces

Summary

Specifies a non-blocking enclosure

Description

See the BLOCKING SURFACES command for a complete description.

Output Database Name

Syntax

Output Database Name *{=}* *Filename* In *{ascii | binary | pnetcdf}* Format

Summary

Specify filename to write viewfactors to

Description

This line command activates the output of the view factors calculated during a simulation. The file name is provided by the user, and one of three file formats is specified. See the "DATABASE NAME" line command description for more detail on the file formats.

Parameter	Value	Default
{=}	{= | are | is}	–
Filename	string	–
VFfileFormat	{ascii | binary | pnetcdf}	–

Overlapping Enclosure

Syntax

Overlapping Enclosure *{=} {false | no | off | on | true | yes}*

Summary

Designates this enclosure as overlapping other enclosures.

Description

On occasion one might want to define an enclosure radiation problem as a superposition of several enclosures. In this case one must designate one of the enclosures as overlapping the others in order to distinguish it from the other enclosures. This command adds a qualifier to the enclosure definition to enable making this distinction possible.

Parameter	Value	Default
{=}	{= | are | is}	–
Bool	{false | no | off | on | true | yes}	–

Partial Enclosure Area

Syntax

Partial Enclosure Area *{=} a*

Summary

Constant value for the partial enclosure area associated with this enclosure radiation flux boundary condition.

Parameter	Value	Default
{=}	{= | are | is}	–
a	real	–

Partial Enclosure Area Subroutine

Syntax

Partial Enclosure Area Subroutine *{=} FName*

Summary

User-defined function name for the partial enclosure area associated with this enclosure radiation flux boundary condition.

Parameter	Value	Default
{=}	{= | are | is}	–
FName	string	–

Partial Enclosure Area Time Function

Syntax

Partial Enclosure Area Time Function *{=}* *FName*

Summary

Time-dependent function name for the partial enclosure area associated with this enclosure radiation flux boundary condition.

Parameter	Value	Default
{=}	{= | are | is}	–
FName	string	–

Partial Enclosure Emissivity

Syntax

Partial Enclosure Emissivity *{=}* *e*

Summary

Constant value for the partial enclosure emissivity associated with this enclosure radiation flux boundary condition.

Parameter	Value	Default
{=}	{= | are | is}	–
e	real	–

Partial Enclosure Emissivity Subroutine

Syntax

Partial Enclosure Emissivity Subroutine *{=}* *FName*

Summary

User-defined function name for the partial enclosure emissivity associated with this enclosure radiation flux boundary condition.

Parameter	Value	Default
{=}	{= | are | is}	–
FName	string	–

Partial Enclosure Emissivity Time Function

Syntax

Partial Enclosure Emissivity Time Function *{=}* *FName*

Summary

Time-dependent function name for the partial enclosure emissivity associated with this enclosure radiation flux boundary condition.

Parameter	Value	Default
{=}	{= | are | is}	–
FName	string	–

Partial Enclosure Flux Output

Syntax

Partial Enclosure Flux Output *VariableName*

Summary

Output the flux associated with the partial enclosure.

Description

During the simulation the flux associated with the partial enclosure is output to the ExodusII file as a global variable.

Parameter	Value	Default
VariableName	string	–

Partial Enclosure Irradiance Output

Syntax

Partial Enclosure Irradiance Output *VariableName*

Summary

Output the irradiance associated with the partial enclosure.

Description

During the simulation the irradiance associated with the partial enclosure is output to the ExodusII file as a global variable.

Parameter	Value	Default
VariableName	string	–

Partial Enclosure Radiosity Output

Syntax

Partial Enclosure Radiosity Output *VariableName*

Summary

Output the radiosity associated with the partial enclosure.

Description

During the simulation the radiosity associated with the partial enclosure is output to the ExodusII file as a global variable.

Parameter	Value	Default
VariableName	string	–

Partial Enclosure Temperature

Syntax

Partial Enclosure Temperature *{=}* *t*

Summary

Constant value for the partial enclosure temperature associated with this enclosure radiation flux boundary condition.

Parameter	Value	Default
{=}	{= | are | is}	–
t	real	–

Partial Enclosure Temperature Subroutine

Syntax

Partial Enclosure Temperature Subroutine *{=}* *FName*

Summary

User-defined function name for the partial enclosure temperature associated with this enclosure radiation flux boundary condition.

Parameter	Value	Default
{=}	{= | are | is}	–
FName	string	–

Partial Enclosure Temperature Time Function

Syntax

Partial Enclosure Temperature Time Function *{=}* *FName*

Summary

Time-dependent function name for the partial enclosure temperature associated with this enclosure radiation flux boundary condition.

Parameter	Value	Default
{=}	{= | are | is}	–
FName	string	–

Preprocess Enclosures

Syntax

Preprocess Enclosures [*Options. . .*]

Summary

Preprocess enclosures subject to options: ORIGINAL_FACES, ENCLOSURE_FACETS, ENCLOSURE_FACET_EDGES, CONTACT_FACETS, CONTACT_FACET_EDGES and COARSE_MESH. When invoked at Region preprocessing is performed and the simulation will terminate.

Description

For debug and inspection purposes the surfaces defined in the enclosure definitions will be processed, then output to the exodus database before terminating the simulation. If no options are chosen the default is to select all the available information. Options are: ORIGINAL_FACES, ENCLOSURE_FACETS, ENCLOSURE_FACET_EDGES, CONTACT_FACETS, CONTACT_FACET_EDGES and COARSE_MESH. Facets are output as triangle elements, edges as bars and faces as shell elements.

Radiosity Database Name

Syntax

Radiosity Database Name *{=}* *Filename*

Summary

Specify filename to write enclosure rowsum error and radiosity results to

Description

This line command activates the output of the view factor rowsum error calculated during a simulation. Additionally, this file will be used to also output the radiosity and flux values during subsequent calls to the radiosity solver. The file name is provided by the user and the output format is ExodusII. This file is useful when the analyst wants to where in the enclosure the maximum errors are occurring. The rowsum error is given by [1-sum_{j = 1}^{N_{facets}} F_{ij}] for each row i in the view-factor matrix (which corresponds to a face in the enclosure). Use of this option will negatively affect performance and is intended for use by developers.

Parameter	Value	Default
{=}	{= | are | is}	–
Filename	string	–

Rowsum Database Name

Syntax

Rowsum Database Name *{=}* *Filename*

Summary

Specify filename to write enclosure rowsum error results to

Description

This line command activates the output of the view factor rowsum error calculated during a simulation. The file name is provided by the user and the output format is ExodusII. This file is useful when the analyst wants to where in the enclosure the maximum errors are occurring. The rowsum error is given by [1-sum_{j = 1}^{N_{facets}} F_{ij}] for each row i in the view-factor matrix (which corresponds to a face in the enclosure).

Parameter	Value	Default
{=}	{= | are | is}	–
Filename	string	–

Topology Database Name

Syntax

Topology Database Name *{=}* *Filename*

Summary

Specify filename to write enclosure topology to

Description

This line command activates the output of the enclosure topology, taking into account all specified symmetry. The file name is provided by the user and the output format is ExodusII.

Parameter	Value	Default
{=}	{= | are | is}	–
Filename	string	–

Use Banded Wavelength Model

Syntax

Use Banded Wavelength Model *ModelName*

Summary

Use a specified banded wavelength model for use in enclosure radiation.

Description

Requests that a specific banded wavelength model named ModelName, be associated with an enclosure, where the ModelName is defined outside of the enclosure definition block. Only one BANDED WAVELENGTH MODEL can be used for an enclosure.

Parameter	Value	Default
ModelName	string	–

Use Dash Enclosures

Syntax

Use Dash Enclosures [*type tol {=} Params. . .*]

Summary

Perform DASH contact search on the enclosure surfaces.

Description

Performing a DASH contact search on the enclosure surfaces will prevent any faces in contact from being processed as part of an enclosure. Optionally one may define a relative tolerance based upon the characteristic element length, or absolute tolerances in positive and negative directions from the enclosure surface.

Use Radiosity Solver

Syntax

Use Radiosity Solver *Param0*

Summary

Specifies which radiosity solver to use.

Description

This line command is used to identify the radiosity solver for use with this enclosure radiation problem. The details of the radiosity solver are defined using the RADIOSITY SOLVER command block. The parameter for this line command is the parameter used with the RADIOSITY SOLVER block command.

Parameter	Value	Default
Param0	string	–

Use Skinned Shells On Surface

Syntax

Use Skinned Shells On Surface *SkinnedSurfaceList. . .*

Summary

Automatically created skinned shells for the specified surfaces and use them instead of the surface for enclosure radiation calculations

Description

Automatically created skinned shells for the specified surfaces and use them instead of the surface for enclosure radiation calculations. This may be required, for example, if the block attached to the enclosure surface is undergoing mesh refinement and you do not want to re-do the view factor calculation every time the surface mesh is modified. This option will create a shell block that's a copy of the specified surface but does not participate in mesh refinement. Temperatures are transferred to the shell and fluxes returned to the source surface using a contact algorithm. For purposes of error estimation, be aware that using

this capability means that the view factor calculation on this surface will always be at the resolution of the mesh on the source surface in the input mesh file.

Parameter	Value	Default
SkinnedSurfaceList	string. . .	–

Use Toggle Block

Syntax

Use Toggle Block *ToggleName* [*{on} ElementBlockList. . .*]

Summary

Specification for toggling entities in the computational model based on Toggle Block parameters. When used at the region level, the list of element blocks to be toggled must be provided. Otherwise a listing of entities is not needed as the Toggle Block will be associated with the command line or the enclosing command block.

Parameter	Value	Default
ToggleName	string	–

Use Viewfactor Calculation

Syntax

Use Viewfactor Calculation *Param0*

Summary

Specifies which view factor calculation to use.

Description

This line command is used to specify the name of a method for calculating view factors defined using the VIEWFACTOR CALCULATION command block. The parameter for this line command is the parameter used to name the VIEWFACTOR CALCULATION block command.

Parameter	Value	Default
Param0	string	–

Use Viewfactor Smoothing

Syntax

Use Viewfactor Smoothing *Param0*

Summary

Specifies which view factor smoother to use.

Description

This line command is used to specify the name of a method for smoothing view factors defined using the VIEWFACTOR SMOOTHING command block. The parameter for this line command is the parameter used with the VIEWFACTOR SMOOTHING block command.

Parameter	Value	Default
Param0	string	–

Viewfactor Update

Syntax

Viewfactor Update *{=} {affected_area | continuous | freeze | interval | only_if_geometry_changed | standard | times}* [Using *Params. . .*]

Summary

Specify the viewfactor re-compute strategy.

Description

The view factor calculation oftentimes dominates the computational cost of a thermal analysis. Thus for cases in which only the geometry changes due to surface deformation and the analyst views the change to be small enough so as to not affect the view factors greatly, one can select one of the strategies below for update of the view factors.

Warning: This command has only been enabled to work with models with mesh motion and will not work with standard element death.

standard is the default re-compute strategy, update on mesh change.

freeze means the view factor update is deactivated for subsequent updates. Thus the first view factor calculation determines the view factors used throughout the simulation.

continuous means the view factors will be computed every time step of a transient simulation.

affected_area is relevant only when used with Element Death. For every view factor calculation the enclosure surface area is computed and stored as a reference area. At subsequent solution steps the area corresponding to elements which are candidates for Element Death is accumulated. The radio of candidate death element area to the original area is used as the criteria of whether Element Death and a new view factor update will occur. If *affected_area* is chosen and Element Death is not present the re-compute strategy will revert to *standard*. Here the *Params* value is the ratio of affected area to total area for which

element death can occur.

Warning: The *affected_area* option has not yet been enabled for use and will not control the viewfactor update frequency.

interval specifies the time interval between view factor updates. Here *Params* is a single value time increment.

times selects specific times at which the view factors are recalculated. Here *Params* can be several discrete time values.

For *interval* and *times*, one can also select a time at which view factor updates are considered using the *view factor update start time* command line.

Parameter	Value	Default
{=}	{= | are | is}	–
UpdateMethod	{affected_area | continuous | freeze | interval | only_if_geometry_changed | standard | times}	STANDARD

Viewfactor Update Start Time

Syntax

Viewfactor Update Start Time Is *Start_time*

Summary

Specify start time for possible re-compute of viewfactors.

Description

Specify the time after which viewfactor re-compute can occur. Command is active only for UPDATE FACTOR method = TIMES or INTERVAL.

Parameter	Value	Default
Start_time	real	0.0

Banded Wavelength Model

Scope

Aria Region, Equation System, Explicit Equation System, Root Finder Equation System

Summary

This command block specifies the wavelength distribution for a banded wavelength model.

Description

Defines a piecewise constant banded wavelength discretization model of the thermal wavelength spectrum and associate it with name ModelName. The same ModelName can be used for different boundary conditions.

```
begin Banded Wavelength Model ModelName

   Band BandName {=} LowValue HighValue

end Banded Wavelength Model ModelName
```

Line Commands

Band

Syntax

Band *BandName {=} LowValue HighValue*

Summary

Specify the range of a wavelength band in microns μm for a surface or collection of surfaces undergoing radiative heat transfer.

Description

Defines the upper and lower limits of a wavelength band in microns μm. Upper and lower limit values can intersect those of adjacent bands but cannot overlap into previously a defined BAND.

Parameter	Value	Default
BandName	string	–
{=}	{= | are | is}	–
LowValue	real	0.0
HighValue	real	REAL_MAX

Band

Scope

Enclosure Definition

Summary

This nested command block specifies the emissivity distribution for the surfaces in the enclosure for the given band.

Description

Defines the emissivity distribution for the surfaces and/or default values for the band with name bandName.

```
begin Band BandName

   Emissivity {=} Value [ On SurfaceName  ]

   Emissivity Function {=} FunctionName [ On SurfaceName  ]

   Emissivity Subroutine {=} MySub [ On SurfaceName  ]

   Emissivity Time Function {=} FunctionName [ On SurfaceName  ]

end Band BandName
```

Line Commands

Emissivity

Syntax

Emissivity *{=}* *Value* [On *SurfaceName*]

Summary

Sets a constant value of emissivity for a defined surface. If the optional parameters are not included, then this is the default emissivity for the enclosure. Otherwise it is only applied to the indicated surface.

Note: If a surface is called out more than once, the emissivity definition is overwritten: last one in wins.

Parameter	Value	Default
{=}	{= | are | is}	–
Value	real	–

Emissivity Function

Syntax

Emissivity Function *{=}* *FunctionName* [On *SurfaceName*]

Summary

Sets a emissivity function for a defined surface. If the optional parameters are not included, then this is the default emissivity for the enclosure. Otherwise it is only applied to the indicated surface.

Parameter	Value	Default
{=}	{= | are | is}	–
FunctionName	string	–

Emissivity Subroutine

Syntax

Emissivity Subroutine *{=}* *MySub* [On *SurfaceName*]

Summary

Sets a emissivity user subroutine for a defined surface. If the optional parameters are not included, then this is the default emissivity for the enclosure. Otherwise it is only applied to the indicated surface.

Also, the software supports using locally scoped user data for most user subroutines, but I haven't figured out a syntax for it here yet. So it is not yet supported. If you need to get data into this subroutine, use the region's "REAL DATA" and "INTEGER DATA" line commands.

Note: If a surface is called out more than once, the emissivity definition is overwritten: last one in wins.

Parameter	Value	Default
{=}	{= | are | is}	–
MySub	string	–

Emissivity Time Function

Syntax

Emissivity Time Function *{=}* *FunctionName* [On *SurfaceName*]

Summary

Sets a emissivity function of time for a defined surface. If the optional parameters are not included, then this is the default emissivity for the enclosure. Otherwise it is only applied to the indicated surface.

Parameter	Value	Default
{=}	{= | are | is}	–
FunctionName	string	–

7.27 Ray Trace Source

Raytrace Source

Scope

Aria Region, Equation System, Explicit Equation System, Root Finder Equation System

Summary

Specification of ray source to be used as part of a ray tracing boundary condition

```
begin Raytrace Source Value

   Absorption Expression [For Band Band] {=} Value

   Absorption Model {=} {fresnel | specular}

   Banded Wavelength Model {=} Value

   Direction Function {=} Value

   Discretization {=} {monte_carlo | radial}

   Exponent {=} Value

   Exponential Factor {=} Value

   Flux Function [For Band Band] {=} Value

   Maximum Intersections {=} Value

   Minimum Power Ratio {=} Value

   Number Of Circular Segments {=} Value

   Number Of Radial Segments {=} Value

   Number Of Rays {=} Value

   Parallel Type {=} {ray | scene}

   Path Function {=} Value

   Power Function [For Band Band] {=} Value
```

```
   Radius {=} Value

   Type {=} {circular | circular_gaussian}

end Raytrace Source Value
```

Line Commands

Absorption Expression

Syntax

Absorption Expression [For Band *Band*] *{=}* *Value*

Summary

Specify the expression to use to calculate the surface absorption properties

Parameter	Value	Default
Band	string	–
{=}	{= | are | is}	–
Value	string	–

Absorption Model

Syntax

Absorption Model *{=}* *{fresnel | specular}*

Summary

Set the method for calculating ray interactions with surfaces. Valid values are ‘Specular’ and ‘Fresnel’

Parameter	Value	Default
{=}	{= | are | is}	–
Value	{fresnel | specular}	–

Banded Wavelength Model

Syntax

Banded Wavelength Model *{=}* *Value*

Summary

Specify a banded wavelength model to use for banded radiation

Parameter	Value	Default
{=}	{= | are | is}	–
Value	string	–

Direction Function

Syntax

Direction Function *{=}* *Value*

Summary

Specify the user function giving the 3 components of the travel direction for the rays as a function of time

Parameter	Value	Default
{=}	{= | are | is}	–
Value	string	–

Discretization

Syntax

Discretization *{=}* *{monte_carlo \| radial}*

Summary

Set the method for discretizing the source into rays. Valid values are 'Monte Carlo' and 'Radial'

Parameter	Value	Default
{=}	{= | are | is}	–
Value	{monte_carlo | radial}	–

Exponent

Syntax

Exponent *{=}* *Value*

Summary

Set the exponent for the Circular_Gaussian type. An exponent of 2 gives a standard Gaussian and is the default

Parameter	Value	Default
{=}	{= | are | is}	–
Value	integer	–

Exponential Factor

Syntax

Exponential Factor *{=}* *Value*

Summary

Set the exponential factor. Required for the Circular_Gaussian type

Parameter	Value	Default
{=}	{= | are | is}	–
Value	real	–

Flux Function

Syntax

Flux Function [For Band *Band*] *{=}* *Value*

Summary

Specify the user function giving the flux of the circular sources as a function of time

Parameter	Value	Default
Band	string	–
{=}	{= | are | is}	–
Value	string	–

Maximum Intersections

Syntax

Maximum Intersections *{=}* *Value*

Summary

Set the number of times that a ray may intersect surfaces before it is no longer tracked. Default is 10

Parameter	Value	Default
{=}	{= | are | is}	–
Value	integer	–

Minimum Power Ratio

Syntax

Minimum Power Ratio *{=}* *Value*

Summary

Set the minimum fraction of initial power that a ray may have before it is no longer tracked. Default is 0.001

Parameter	Value	Default
{=}	{= | are | is}	–
Value	real	–

Number Of Circular Segments

Syntax

Number Of Circular Segments *{=}* *Value*

Summary

Set the number of segments to discretize the circle circumference into at the innermost radial segment. Required for the radial discretization type

Parameter	Value	Default
{=}	{= | are | is}	–
Value	integer	–

Number Of Radial Segments

Syntax

Number Of Radial Segments *{=}* *Value*

Summary

Set the number of segments to discretize the circle radius into. Required for the radial discretization type

Parameter	Value	Default
{=}	{= | are | is}	–
Value	integer	–

Number Of Rays

Syntax

Number Of Rays *{=}* *Value*

Summary

Set the number of rays to be generated by the circular sources. Required for the Monte Carlo discretization type

Parameter	Value	Default
{=}	{= | are | is}	–
Value	integer	–

Parallel Type

Syntax

Parallel Type *{=}* *{ray | scene}*

Summary

Set the parallel decomposition method for the ray tracer. Valid values are ‘Ray’ and ‘Scene’

Parameter	Value	Default
{=}	{= | are | is}	–
Value	{ray | scene}	–

Path Function

Syntax

Path Function *{=}* *Value*

Summary

Specify the user function giving the x, y, and z position of the circular sources as a function of time

Parameter	Value	Default
{=}	{= | are | is}	–
Value	string	–

Power Function

Syntax

Power Function [For Band *Band*] *{=}* *Value*

Summary

Specify the user function giving the power of the circular sources as a function of time

Parameter	Value	Default
Band	string	–
{=}	{= | are | is}	–
Value	string	–

Radius

Syntax

Radius *{=}* *Value*

Summary

Set the radius for the circular shapes

Parameter	Value	Default
{=}	{= | are | is}	–
Value	real	–

Type

Syntax

Type *{=} {circular | circular_gaussian}*

Summary

Set the type of the ray source. Valid values are 'Circular' and 'Circular_Gaussian'

Parameter	Value	Default
{=}	{= | are | is}	–
Value	{circular | circular_gaussian}	–

7.28 General Chemistry

General Chemistry

Scope

Aria Material

```
begin General Chemistry MechanismName

   Activation Energies {=} Value...

   Activation Temperature {=} Value

   Concentration Exponents For SpeciesName {=} Exponents...

   Deactivation Temperature {=} Value Action

   Deactivation Temperature Overshoot {=} Value

   Deactivation Temperature Rate {=} Value Action

   Enthalpy Of Formation For SpeciesName {=} Value

   Heats Of Reaction {=} Value...

   Newton Temperature Solve {=} Value

   Nucleation Time {=} Value

   Number Of Reactions {=} Value

   Phase Transfer Preserved Variable {=} Value
```

```
   Post Deactivation Gas Release {=} Value

   Post Deactivation Heat Release {=} Value

   Post Deactivation Release Time {=} Value

   Preexponential Factors {=} Value...

   Species Names {=} Value...

   Species Phases {=} Value...

   Species Variable Name {=} Value

   Steric Coefficients {=} Value...

   Stoichiometric Coefficients For SpeciesName {=} Coefficients...

   Temperature Phases {=} Value...

   Universal Gas Constant {=} Value

   begin Reaction ReactionName
   end

end General Chemistry MechanismName
```

Line Commands

Activation Energies

Syntax

Activation Energies *{=} Value...*

Summary

Activation energy, E, for each Arrhenius-rate reaction

Description

The units for this quantity may be whatever you like, as long as it results in a dimensionless quantity when combined with the temperature and universal gas constant.

Parameter	Value	Default
{=}	{= | are | is}	–
Value	real. . .	–

Activation Temperature

Syntax

Activation Temperature *{=}* *Value*

Summary

Activation temperature

Description

This specifies the activation temperature for general chemistry

Parameter	Value	Default
{=}	{= | are | is}	–
Value	real	–

Concentration Exponents For

Syntax

Concentration Exponents For *SpeciesName* *{=}* *Exponents. . .*

Summary

Concentration exponents, μ, for each reactant species for each Arrhenius rate reaction

Description

Values of zero imply that the species is not a reactant in the specified reaction

Parameter	Value	Default
SpeciesName	string	–
{=}	{= | are | is}	–
Exponents	real. . .	–

Deactivation Temperature

Syntax

Deactivation Temperature *{=}* *Value* *Action*

Summary

Deactivation temperature

Description

This specifies the deactivation temperature for general chemistry along with the deactivation action (terminate or continue)

Parameter	Value	Default
{=}	{= | are | is}	–
Value	real	–
Action	string	–

Deactivation Temperature Overshoot

Syntax

Deactivation Temperature Overshoot *{=}* *Value*

Summary

Allowable overshoot for deactivation

Description

This specifies the amount the temperature is allowed to overshoot the specified deactivation temperature without failing the timestep

Parameter	Value	Default
{=}	{= | are | is}	–
Value	real	–

Deactivation Temperature Rate

Syntax

Deactivation Temperature Rate *{=}* *Value* *Action*

Summary

Deactivation temperature rate

Description

This specifies the deactivation temperature rate for general chemistry along with the deactivation action (terminate or continue)

Parameter	Value	Default
{=}	{= | are | is}	–
Value	real	–
Action	string	–

Enthalpy Of Formation For

Syntax

Enthalpy Of Formation For *SpeciesName* *{=}* *Value*

Summary

Enthalpy of formation for each species

Description

This specifies the enthalpy of formation at the reference temperature for each species. This should be in a consistent mass or mole-based unit system with the rest of the mechanism.

Parameter	Value	Default
SpeciesName	string	–
{=}	{= | are | is}	–
Value	real	–

Heats Of Reaction

Syntax

Heats Of Reaction *{=}* *Value. . .*

Summary

Heat of reaction for each reaction

Description

This specifies the heat of reaction at the reference temperature for each reaction. This should be in a consistent mass or mole-based unit system with the rest of the mechanism. Positive values represent an endothermic reaction and negative values represent an exothermic reaction.

Parameter	Value	Default
{=}	{= | are | is}	–
Value	real. . .	–

Newton Temperature Solve

Syntax

Newton Temperature Solve *{=}* *Value*

Summary

Newton Temperature Solve

Description

If set to true, use Newton's method to solve for the temperature. Otherwise a constant heat capacity is assumed when calculating temperature.

Parameter	Value	Default
{=}	{= | are | is}	–
Value	string	–

Nucleation Time

Syntax

Nucleation Time *{=}* *Value*

Summary

Nucleation time to be used in rate calculations

Description

If specified then all reaction rates will be scaled by $\frac{1}{2}\left(1 + tanh\left(\frac{t-t_{nuc}}{t_{nuc}}\right)\right)$.

Parameter	Value	Default
{=}	{= | are | is}	–
Value	real	–

Number Of Reactions

Syntax

Number Of Reactions *{=}* *Value*

Summary

Number of reactions in the chemical mechanism

Parameter	Value	Default
{=}	{= | are | is}	–
Value	integer	–

Phase Transfer Preserved Variable

Syntax

Phase Transfer Preserved Variable *{=}* *Value*

Summary

Use isothermal phase transfers

Description

Options are ‘temperature’ or ‘enthalpy’. This allows you to use temperature-preserving transfers of material from one phase to the other, rather than using enthalpy-preserving transfers (the default).

Parameter	Value	Default
{=}	{= | are | is}	–
Value	string	–

Post Deactivation Gas Release

Syntax

Post Deactivation Gas Release *{=}* *Value*

Summary

Gas release after deactivation

Description

Specifies a uniform energy release to be applied after deactivation. The units of this value are moles per volume (e.g. mol/m3)

Parameter	Value	Default
{=}	{= | are | is}	–
Value	real	–

Post Deactivation Heat Release

Syntax

Post Deactivation Heat Release *{=}* *Value*

Summary

Heat release after deactivation

Description

Specifies a uniform energy release to be applied after deactivation. The units of this value are energy per volume (e.g. J/m3)

Parameter	Value	Default
{=}	{= | are | is}	–
Value	real	–

Post Deactivation Release Time

Syntax

Post Deactivation Release Time *{=}* *Value*

Summary

Time for heat and gas release after deactivation

Description

Specifies the time to use for the uniform heat and/or gas release terms.

Parameter	Value	Default
{=}	{= | are | is}	–
Value	real	–

Preexponential Factors

Syntax

Preexponential Factors *{=}* *Value. . .*

Summary

Pre-exponential factor, A, for each Arrhenius-rate reaction

Description

Please note that this is the actual Arrhenius pre-exponential factor, A, and not the logarithm of the value (as was expected in the previous CHEMEQ capability).

Parameter	Value	Default
{=}	{= | are | is}	–
Value	real. . .	–

Species Names

Syntax

Species Names *{=}* *Value. . .*

Summary

Names of all species participating in chemical mechanism

Description

All of these names must be valid species defined elsewhere in the Material block (or in material phase sub-blocks).

Parameter	Value	Default
{=}	{= | are | is}	–
Value	string. . .	–

Species Phases

Syntax

Species Phases *{=}* *Value. . .*

Summary

Name of the material phases for each chemical species

Description

Provide the full name of the material phase for each chemical species, in the same order as the species are listed. If this command is omitted, then all species are assumed to not have a material phase.

Parameter	Value	Default
{=}	{= | are | is}	–
Value	string. . .	–

Species Variable Name

Syntax

Species Variable Name *{=}* *Value*

Summary

Names of the input species variable to be used in rate calculations

Description

If the chemical mechanism is formulated on a molar basis then this field must be in units of molar concentration (e.g. mol_i/m^3), and if the mechanism is formulated on a mass basis then this field must be in units of mass concentration (e.g. kg_i/m^3). If the solution variable is neither of these (e.g. mass fraction, mole fraction, etc.), then a material model to compute the required concentrations from the available solution variable must be provided.

Parameter	Value	Default
{=}	{= | are | is}	–
Value	string	–

Steric Coefficients

Syntax

Steric Coefficients *{=}* *Value...*

Summary

Steric coefficient exponent for each reaction

Description

The steric coefficient has the form T^{β}, where β is the provided value. Zero will be assumed as the default for all reactions if this optional line command is not provided.

Parameter	Value	Default
{=}	{= | are | is}	–
Value	real...	–

Stoichiometric Coefficients For

Syntax

Stoichiometric Coefficients For *SpeciesName* *{=}* *Coefficients...*

Summary

Stoichiometric coefficients, ν, for each reactant species for each reaction

Description

These coefficients must sum to zero for each reaction if using a mass-based formulation.

Parameter	Value	Default
SpeciesName	string	–
{=}	{= | are | is}	–
Coefficients	real. . .	–

Temperature Phases

Syntax

Temperature Phases *{=}* *Value. . .*

Summary

The material phase of the temperature to use for this reaction

Description

In multi-material phase heterogeneous mechanisms, it is required to specify which phase's temperature to use for each reaction.

Parameter	Value	Default
{=}	{= | are | is}	–
Value	string. . .	–

Universal Gas Constant

Syntax

Universal Gas Constant *{=}* *Value*

Summary

Universal gas constant to be used in rate calculations

Description

This is the standard universal gas constant, for use in the exponential term of the Arrhenius rate expression. The units must be consistent with the provided activation energy. If specifying an activation temperature (E/R_u) is preferred, then this can be set to 1 and activation temperatures can be provided for the activation energies.

Parameter	Value	Default
{=}	{= | are | is}	–
Value	real	–

Chemistry Solver Parameters For

Scope

Aria Region, Equation System, Explicit Equation System, Root Finder Equation System

Summary

Parameters for the chemistry solver applicable to the previously defined material with chemistry model ModelName. The chemistry solver model produces no independent action since the ModelName must first appear in a SRC term for an ENERGY or SPECIES equation before it be applied.

```
begin Chemistry Solver Parameters For ModelName

   Absolute Tolerance {=} absTol

   Chemistry Solver Algorithm {=} algorithmType

   Maximum Substeps {=} maxSubSteps

   Minimum Step Size {=} minStepSize

   Ode Solver {=} odeSolver...

   Relative Tolerance {=} relTol

end Chemistry Solver Parameters For ModelName
```

Line Commands

Absolute Tolerance

Syntax

Absolute Tolerance *{=}* *absTol*

Summary

Specifies the absolute solution tolerance for the ODE solver

Parameter	Value	Default
{=}	{= | are | is}	–
absTol	real	1e-12

Chemistry Solver Algorithm

Syntax

Chemistry Solver Algorithm *{=}* *algorithmType*

Summary

This command specifies the coupling algorithm between the chemistry model and transport equations

Parameter	Value	Default
{=}	{= | are | is}	–
algorithmType	string	COUPLED

Maximum Substeps

Syntax

Maximum Substeps *{=}* *maxSubSteps*

Summary

Specifies the maximum number of chemistry substeps per global time step.

Parameter	Value	Default
{=}	{= | are | is}	–
maxSubSteps	integer	100000

Minimum Step Size

Syntax

Minimum Step Size *{=}* *minStepSize*

Summary

Specifies the minimum integration step size.

Parameter	Value	Default
{=}	{= | are | is}	–
minStepSize	real	1e-30

Ode Solver

Syntax

Ode Solver *{=}* *odeSolver. . .*

Summary

This command selects the ODE solver to use for a segregated solve. For available options, refer to the ‘ODE Solvers’ section of this chapter.

Parameter	Value	Default
{=}	{= | are | is}	–
odeSolver	string. . .	CVODE

Relative Tolerance

Syntax

Relative Tolerance *{=}* *relTol*

Summary

Specifies the relative solution tolerance for the ODE solver

Parameter	Value	Default
{=}	{= | are | is}	–
relTol	real	1e-6

7.29 Heat Transfer Correlation Coefficient

Heat Transfer Correlation Coefficient

Scope

Aria Region, Equation System, Explicit Equation System, Root Finder Equation System

Summary

Specified heat transfer correlation coefficient model.

Description

Enables the user to specify a correlation model for use with computing heat transfer coefficients in conjunction with advective bar networks.

```
begin Heat Transfer Correlation Coefficient COEFF name

  Apply Gnielinski Film Gradient Correction [With Exponent {=} Exponent]

  Apply Hausen Entrance Effect Correction [Maximum Correction {=}␣
↪MaxCorrection]

  Apply Wall Thermal Resistance = Scalar_String_Function F = string_
↪function

  Characteristic Length {=} Dh

  Compute {annulus diameter ratio | characteristic length | density␣
↪| fluid temperature | fluid velocity | friction factor | kinematic␣
↪viscosity | prandtl number | reynolds number | wall prandtl number |␣
↪wall temperature} [ Model {=} Value...  ]

  Density {=} rho
```

```
   Entrance Length {=} De

   Expansion Coefficient {=} beta

   Fluid Phase {=} {gas | liquid}

   Friction Factor {=} f

   Gravitational Constant {=} g

   Kinematic Viscosity {=} nu

   Laminar Correlation {=} Type {1 | 14 | 15 | 18 | 2 | 21 | 23 | 25 | 26␣
↪| 27 | 28 | 3 | 34 | 35 | 36 | 37 | 38 | 39 | 4 | 5 | 58 | 59 | 60 | 61␣
↪| 62 | 63 | 64 | 65 | 72 | 73 | 74}

   Laminar Correlation = Scalar_String_Function F = string_function

   Porosity {=} phi

   Prandtl Number {=} Pr

   Reynolds Number {=} Re

   Turbulent Correlation {=} Type {1 | 14 | 15 | 18 | 2 | 21 | 23 | 25 |␣
↪26 | 27 | 28 | 3 | 34 | 35 | 36 | 37 | 38 | 39 | 4 | 5 | 58 | 59 | 60 |␣
↪61 | 62 | 63 | 64 | 65 | 72 | 73 | 74}

   Turbulent Correlation = Scalar_String_Function F = string_function

   Temperature {=} t

   Transition Reynolds Number {=} Rex

   Velocity {=} v

   Wall Angle {=} theta

   Wall Length {=} l

   Wall Temperature {=} Tw

end Heat Transfer Correlation Coefficient COEFF name
```

Line Commands

Apply Gnielinski Film Gradient Correction

Syntax

Apply Gnielinski Film Gradient Correction [With Exponent *{=}* *Exponent*]

Summary

Enable a Gnielinski film temperature gradient correction factor to account for variation of fluid properties with temperature when computing the pipe and annular flow correlation heat transfer coefficients (Types 23, 25, 72-74).

Description

For gas phase convection the heat transfer coefficient correlations include a correction factor of the form $\left(\frac{T}{T_w}\right)^n$.

For gas cooling $\left(\frac{T}{T_w}\right) > 1.0$ $n = 0$.

For gas heating $0.5 < \left(\frac{T}{T_w}\right) < 1.0$, n varies for different gases but $n = 0.45$ is generally recommended. For CO_2 and steam $n = 0.15$.

For liquids the correction factor is a function of bulk fluid and wall Prandtl number $\left(\frac{Pr_b}{Pr_w}\right)^{0.11}$.

Parameter	Value	Default
{=}	{= | are | is}	–
Exponent	real	NONE

Apply Hausen Entrance Effect Correction

Syntax

Apply Hausen Entrance Effect Correction [Maximum Correction *{=}* *MaxCorrection*]

Summary

Enable the Hausen entrance effect correction factor for pipe and annular flow correlation heat transfer coefficients (Types 23, 25, 72-74).

Description

Applies a correction factor of the form $\left(1 + \frac{d_H}{L}^{\frac{2}{3}}\right)$ to the heat transfer coefficient where d_H is the hydraulic diameter and L is the distance from the pipe or annulus entrance to the point the flux is applied. By default the maximum value of the correction is 10, it can optionally be set using the MaxCorrection parameter in this command line. For $L < L_{min}$ the correction factor is not

applied. This correction factor may only be applied when the correlation is being used with an advective bar model to represent the fluid flow.

Parameter	Value	Default
{=}	{= | are | is}	–
MaxCorrection	real	10

Apply Wall Thermal Resistance

Syntax

Apply Wall Thermal Resistance = Scalar_String_Function F = *string_function*

Summary

Model a thin wall using a wall thermal resistance specified as a scalar string function. The thermal resistance will be applied in series

Parameter	Value	Default
string_function	"string"	–

Characteristic Length

Syntax

Characteristic Length *{=} Dh*

Summary

Specifies the characteristic length for parameter computations such as Reynolds Number, etc.

Parameter	Value	Default
{=}	{= | are | is}	–
Dh	real	–

Compute

Syntax

Compute *{annulus diameter ratio | characteristic length | density | fluid temperature | fluid velocity | friction factor | kinematic viscosity | prandtl number | reynolds number | wall prandtl number | wall temperature}* [Model *{=} Value. . .*]

Summary

Specifies that model parameter be computed through such means as extraction, generic form, specific model, etc. Current options for the parameters and associated models include

CHARACTERISTIC LENGTH: {Hydraulic Diameter | Wetted Perimeter}

FLUID VELOCITY: This should always be set to the name of the advection velocity used in the energy equation. This will most likely be *advection velocity* which is the internal variable used in Aria but may be set manually to anything else such as a user defined velocity field. FRICTION FACTOR: {Smooth Tube}

Parameter	Value	Default
ComputeCorrParam	{annulus diameter ratio | characteristic length | density | fluid temperature | fluid velocity | friction factor | kinematic viscosity | prandtl number | reynolds number | wall prandtl number | wall temperature}	–

Density

Syntax

Density *{=}* *rho*

Summary

Specify a density to either compute or specify by value for usage with a correlation convection coefficient.

Parameter	Value	Default
{=}	{= | are | is}	–
rho	real	–

Entrance Length

Syntax

Entrance Length *{=}* *De*

Summary

Specify an entrance length for usage with a correlation convection coefficient.

Parameter	Value	Default
{=}	{= | are | is}	–
De	real	–

Expansion Coefficient

Syntax

Expansion Coefficient *{=}* *beta*

Summary

Specify an expansion coefficient by value for usage with a correlation convection coefficient.

Parameter	Value	Default
{=}	{= | are | is}	–
beta	real	–

Fluid Phase

Syntax

Fluid Phase *{=} {gas | liquid}*

Summary

Specify the fluid phase for usage with a correlation convection coefficient.

Parameter	Value	Default
{=}	{= | are | is}	–
FluidPhase	{gas | liquid}	GAS

Friction Factor

Syntax

Friction Factor *{=} f*

Summary

Specify a friction factor to either compute or specify by value for usage with a correlation convection coefficient.

Parameter	Value	Default
{=}	{= | are | is}	–
f	real	–

Gravitational Constant

Syntax

Gravitational Constant *{=} g*

Summary

Specify a gravitational constant by value for usage with a correlation convection coefficient.

Parameter	Value	Default
{=}	{= | are | is}	–
g	real	–

Kinematic Viscosity

Syntax

Kinematic Viscosity *{=} nu*

Summary

Specify a kinematic viscosity to either compute or specify by value for usage with a correlation convection coefficient.

Parameter	Value	Default
{=}	{= | are | is}	–
nu	real	–

Laminar Correlation

Syntax

Laminar Correlation *{=}* Type *{1 | 14 | 15 | 18 | 2 | 21 | 23 | 25 | 26 | 27 | 28 | 3 | 34 | 35 | 36 | 37 | 38 | 39 | 4 | 5 | 58 | 59 | 60 | 61 | 62 | 63 | 64 | 65 | 72 | 73 | 74}*

Summary

Specifies the laminar correlation model.

Parameter	Value	Default
{=}	{= | are | is}	–
CorrelationType	{1 | 14 | 15 | 18 | 2 | 21 | 23 | 25 | 26 | 27 | 28 | 3 | 34 | 35 | 36 | 37 | 38 | 39 | 4 | 5 | 58 | 59 | 60 | 61 | 62 | 63 | 64 | 65 | 72 | 73 | 74}	–

Laminar Correlation

Syntax

Laminar Correlation = Scalar_String_Function F = *string_function*

Summary

Define Nusselt number using a scalar string function

Parameter	Value	Default
string_function	"string"	–

Porosity

Syntax

Porosity *{=}* *phi*

Summary

Specify a Porosity to either compute or specify by value for usage with a correlation convection coefficient.

Parameter	Value	Default
{=}	{= | are | is}	–
phi	real	–

Prandtl Number

Syntax

Prandtl Number *{=}* *Pr*

Summary

Specify a Prandtl number to either compute or specify by value for usage with a correlation convection coefficient.

Parameter	Value	Default
{=}	{= | are | is}	–
Pr	real	–

Reynolds Number

Syntax

Reynolds Number *{=}* *Re*

Summary

Specify a Reynold's number to either compute or specify by value for usage with a correlation convection coefficient.

Parameter	Value	Default
{=}	{= | are | is}	–
Re	real	–

Turbulent Correlation

Syntax

Turbulent Correlation *{=}* Type *{1 | 14 | 15 | 18 | 2 | 21 | 23 | 25 | 26 | 27 | 28 | 3 | 34 | 35 | 36 | 37 | 38 | 39 | 4 | 5 | 58 | 59 | 60 | 61 | 62 | 63 | 64 | 65 | 72 | 73 | 74}*

Summary

Specifies the turbulent correlation model.

Parameter	Value	Default
{=}	{= | are | is}	–
CorrelationType	{1 | 14 | 15 | 18 | 2 | 21 | 23 | 25 | 26 | 27 | 28 | 3 | 34 | 35 | 36 | 37 | 38 | 39 | 4 | 5 | 58 | 59 | 60 | 61 | 62 | 63 | 64 | 65 | 72 | 73 | 74}	–

Turbulent Correlation

Syntax

Turbulent Correlation = Scalar_String_Function F = *string_function*

Summary

Define Nusselt number using a scalar string function

Parameter	Value	Default
string_function	"string"	–

Temperature

Syntax

Temperature *{=}* *t*

Summary

Specify a Fluid Temperature to either compute or specify by value for usage with a correlation convection coefficient.

Parameter	Value	Default
{=}	{= | are | is}	–
t	real	–

Transition Reynolds Number

Syntax

Transition Reynolds Number *{=}* *Rex*

Summary

Specify a transition Reynolds number to either compute or specify by value for usage with a correlation convection coefficient.

Parameter	Value	Default
{=}	{= | are | is}	–
Rex	real	–

Velocity

Syntax

Velocity *{=}* *v*

Summary

Specify a velocity to either compute or specify by value for usage with a correlation convection coefficient.

Parameter	Value	Default
{=}	{= | are | is}	–
v	real	–

Wall Angle

Syntax

Wall Angle *{=}* *theta*

Summary

Specify a wall angle by value for usage with a correlation convection coefficient.

Parameter	Value	Default
{=}	{= | are | is}	–
theta	real	–

Wall Length

Syntax

Wall Length *{=}* *l*

Summary

Specify a wall length by value for usage with a correlation convection coefficient.

Parameter	Value	Default
{=}	{= | are | is}	–
l	real	–

Wall Temperature

Syntax

Wall Temperature *{=}* *Tw*

Summary

Specify a Wall Temperature to either compute or specify by value for usage with a correlation convection coefficient.

Parameter	Value	Default
{=}	{= | are | is}	–
Tw	real	–

Use Correlation Convection Model

Syntax

Use Correlation Convection Model *Name*

Summary

Specifies correlation model for convection coefficient

Parameter	Value	Default
Name	string	–

7.30 Level Set

Level Set Interface

Scope
Aria Region

Summary
Main block for krino level set interface control mechanics.

Description
This is the main block for the level set interface control mechanics. This block specifies a level set interface that is integrated in time using an extension velocity. The plan is that the physics application code will see only this mechanics, although there may be several other supporting mechanics that are plugged in.

```
begin Level Set Interface InterfaceName

   Composite Name {=} Variable

   Distance Variable {=} Variable

   Initial Offset Distance {=} Offset

   Initial Scale Factor {=} Scale

   Max Feature Size On Surfaces {=} Value

   Narrow Band Element Size Multiplier {=} Value

   Narrow Band Width {=} Value

   Perform Initial Redistance {=} {false | no | off | on | true | yes}

   Redistance Method {=} {closest_point | fast_marching}

   Reinitialize Every Step

   Simple Max Feature Size On Surfaces {=} Value

   begin Analytic Initial Condition BlockName
   end

   begin Compute_Surface_Distance SurfaceName
```

```
   end

   begin Motion Specification Motion
   end

end Level Set Interface InterfaceName
```

Line Commands

Composite Name

Syntax

Composite Name *{=}* *Variable*

Summary

Optionally specify a composite level set name which may be used across multiple level sets to define composite phases.

Parameter	Value	Default
{=}	{= | are | is}	–
Variable	string	–

Distance Variable

Syntax

Distance Variable *{=}* *Variable*

Summary

Specify the distance variable name.

Parameter	Value	Default
{=}	{= | are | is}	–
Variable	string	–

Initial Offset Distance

Syntax

Initial Offset Distance *{=}* *Offset*

Summary

Specify an offset distance to be applied to the initial level set. Allows the user to grow or shrink the initialized level set.

Parameter	Value	Default
{=}	{= | are | is}	–
Offset	real	–

Initial Scale Factor

Syntax

Initial Scale Factor *{=}* *Scale*

Summary

Specify an scale factor to be applied to the initial level set. Allows the user to multiple the entire level set by a constant, primarily useful for nested (concentric) level sets. This modifier gets applied after the INITIAL OFFSET DISTANCE, if there is one.

Parameter	Value	Default
{=}	{= | are | is}	–
Scale	real	–

Max Feature Size On Surfaces

Syntax

Max Feature Size On Surfaces *{=}* *Value*

Summary

Specify the maximum feature size on surfaces, features smaller than this will be destroyed near all surfaces. IN BETA TESTING!

Parameter	Value	Default
{=}	{= | are | is}	–
Value	real	–

Narrow Band Element Size Multiplier

Syntax

Narrow Band Element Size Multiplier *{=}* *Value*

Summary

Specify the width of the narrow band width used for the level set field as the given multiplier times the maximum element length scale. If this command is omitted, the distance will be computed throughout the entire domain, but this is not generally scalable in terms of memory or cpu time.

Parameter	Value	Default
{=}	{= | are | is}	–
Value	real	–

Narrow Band Width

Syntax

Narrow Band Width *{=}* *Value*

Summary

Specify the width of the narrow band. This form is deprecated. Use NARROW BAND ELEMENT SIZE MULTIPLIER instead.

Parameter	Value	Default
{=}	{= | are | is}	–
Value	real	–

Perform Initial Redistance

Syntax

Perform Initial Redistance *{=} {false | no | off | on | true | yes}*

Summary

Perform redistancing operation just after initial conditions are applied.

Parameter	Value	Default
{=}	{= | are | is}	–
performInitialRedistance	{false | no | off | on | true | yes}	FALSE

Redistance Method

Syntax

Redistance Method *{=} {closest_point | fast_marching}*

Summary

Specify the algorithm to be used for redistancing.

Parameter	Value	Default
{=}	{= | are | is}	–
RedistanceMethodType	{closest_point | fast_marching}	CLOSEST_POINT

Reinitialize Every Step

Syntax

Reinitialize Every Step

Summary

Specify that this level set interface should be re-initialized every time step. For example this can be combined with an analytic mesh surface initial condition that is moving due to applied displacements to move the CDFEM interface location with the mesh surface.

Simple Max Feature Size On Surfaces

Syntax

Simple Max Feature Size On Surfaces *{=} Value*

Summary

Specify the maximum feature size on surfaces, features smaller than this will be destroyed near all surfaces (simple version) IN BETA TESTING!

Parameter	Value	Default
{=}	{= | are | is}	–
Value	real	–

7.31 Local Coordinate Systems

Local Coordinate System

Scope

Sierra

Summary

Defines the local coordinate system parameters.

Description

This Command Block allows the definition of a coordinate system that can be used to transform values for output or field calculations. For any specific local coordinate system three pieces of information are required for completely defining a coordinate system are the origin, a coordinate axis vector and a point lying on one of the other axes. Using this information a transformation from the local coordinate system to the computational coordinate system can be determined.

```
begin Local Coordinate System Coordinate_System_Name

   Origin {=} Value...

   Point {=} Value...

   Type {=} {cartesian | conical | cylindrical | spherical}

   Vector {=} Value...

end Local Coordinate System Coordinate_System_Name
```

Line Commands

Origin

Syntax

Origin *{=} Value. . .*

Summary

Local coordinate system origin.

Description

Specifies the point as the origin of a local coordinate system. The number of coordinates may be either 2 or 3 depending on the spatial dimension. By default, a value of $z = 0.0$ is used for 2D.

Parameter	Value	Default
{=}	{= | are | is}	–
Value	real. . .	–

Point

Syntax

Point *{=} Value. . .*

Summary

Local coordinate alternative point.

Description

Declare the point that lies on an alternate axis and is in the plane subtended by the specified VECTOR and the vector defined by the point and the origin.

Parameter	Value	Default
{=}	{= | are | is}	–
Value	real. . .	–

Type

Syntax

Type *{=} {cartesian | conical | cylindrical | spherical}*

Summary

Local coordinate system type.

Description

This line command selects the coordinate system type.

Parameter	Value	Default
{=}	{= | are | is}	–
CoordType	{cartesian | conical | cylindrical | spherical}	–

Vector

Syntax

Vector *{=}* *Value. . .*

Summary

Local coordinate axis vector.

Description

Declare the vector that defines one of the coordinate axes. The specific usage depends on the selected coordinate system type.

Parameter	Value	Default
{=}	{= | are | is}	–
Value	real. . .	–

Local Coordinate System

Syntax

Local Coordinate System *{=}* *Mesh Entities*

Summary

Associate coordinate system with mesh entity.

Description

Specify the local coordinate system to be used in conjunction with given element blocks.

Parameter	Value	Default
{=}	{= | are | is}	–
Mesh Entities	string	–

7.32 Mpmd Coupling

Scope

Aria Region

Summary

Allows specification of a coupling with an external application in MPMD mode

```
begin Mpmd Coupling Value

   Coupling Type {=} {aero | radiation}

   begin Aero Coupling Value
   end

   begin Radiation Coupling Value
   end

end Mpmd Coupling Value
```

Line Commands

Coupling Type

Syntax

Coupling Type *{=} {aero | radiation}*

Summary

Set the MPMD coupling type. Valid values are AERO and RADIATION

Parameter	Value	Default
{=}	{= | are | is}	–
Value	{aero | radiation}	–

Aero Coupling

Summary

Allows specification of interface location and properties for coupling with an Aero application.

```
begin Aero Coupling Value

   Coupling Surfaces {=} Value...
```

```
  Implicit Absolute Norm2 Solution Update Tolerance {=} Value [With {dof_
↪amax | dof_rms} Scaling]

  Implicit Norm Inf Solution Update Tolerance {=} Value [With {dof_amax␣
↪| dof_rms} Scaling]

  Implicit Receive Field Relaxation {=} Value

  Implicit Rms Solution Update Tolerance {=} Value [With {dof_amax | dof_
↪rms} Scaling]

  Implicit Send Field Relaxation {=} Value

  Max Number Iterations {=} Value

  Maximum Transfer Clipping Distance {=} Value

  Patch Reconstruction {=} {cubic_bspline | gregory | none}

  Receive Field {=} Value ([Smooth Smoothing Iterations Iterations]  |␣
↪[With Time Interpolation])

  Receive Global {=} Value

  Send Coordinates Field {=} Value

  Send Displacement Field {=} Value

  Send Field {=} Value

  Send Global {=} Value

  Symmetry Plane Tolerance {=} Value

  Symmetry Planes {=} Value...

  Time Step Mode {=} {aero_subcycle | implicit | negotiate_minimum |␣
↪thermal_subcycle}

  Transfer Type {=} {conservative | geometric}

  Unconverged Implicit Steps {=} {accept | reject}

  begin Implicit Tolerances
  end
```

```
   begin Transfer Value
   end

end Aero Coupling Value
```

Line Commands

Coupling Surfaces

Syntax

Coupling Surfaces *{=}* *Value. . .*

Summary

Select the surfaces that are coupled to the aero code.

Parameter	Value	Default
{=}	{= | are | is}	–
Value	string. . .	–

Implicit Absolute Norm2 Solution Update Tolerance

Syntax

Implicit Absolute Norm2 Solution Update Tolerance *{=}* *Value* [With *{dof_amax | dof_rms}* Scaling]

Summary

Specify the absolute L_2 tolerance for mpmd coupled solution update delta.

Description

Specify the convergence tolerance for iterating the mpmd subdomains in terms of an absolute nodal L2 norm on the solution field delta. Optionally specify a scaling to normalize delta by before checking against tolerance.

Parameter	Value	Default
{=}	{= | are | is}	–
Value	real	1.798e+308

Implicit Norm Inf Solution Update Tolerance

Syntax

Implicit Norm Inf Solution Update Tolerance *{=}* *Value* [With *{dof_amax | dof_rms}* Scaling]

Summary

Specify the L_inf tolerance for mpmd coupled solution update delta.

Description

Specify the convergence tolerance for iterating the mpmd subdomains in terms of an L_Inf norm on the solution field delta. Optionally specify a scaling to normalize delta by before checking against tolerance.

Parameter	Value	Default
{=}	{= | are | is}	–
Value	real	1.798e+308

Implicit Receive Field Relaxation

Syntax

Implicit Receive Field Relaxation *{=}* *Value*

Summary

Add relaxation to fields received from coupled app for implicit iterations 2+ $f_{k+1} = \omega f'_{k+1} + (1-\omega) f_k$

Parameter	Value	Default
{=}	{= | are | is}	–
Value	real	1

Implicit Rms Solution Update Tolerance

Syntax

Implicit Rms Solution Update Tolerance *{=}* *Value* [With *{dof_amax | dof_rms}* Scaling]

Summary

Specify the RMS tolerance for mpmd coupled solution update delta.

Description

Specify the convergence tolerance for iterating the mpmd subdomains in terms of the RMS on the solution field delta. Optionally specify a scaling to normalize delta by before checking against tolerance.

Parameter	Value	Default
{=}	{= | are | is}	–
Value	real	1.798e+308

Implicit Send Field Relaxation

Syntax

Implicit Send Field Relaxation *{=}* *Value*

Summary

Add relaxation to fields sent to coupled app for implicit iterations 2+
$f_{k+1} = \omega f'_{k+1} + (1-\omega) f_k$

Parameter	Value	Default
{=}	{= | are | is}	–
Value	real	1

Max Number Iterations

Syntax

Max Number Iterations *{=}* *Value*

Summary

Specify the maximum number of iterations for mpmd coupled simulations.

Description

Aria will stop iterating and accept the solution after the specified number of iterations

Parameter	Value	Default
{=}	{= | are | is}	–
Value	integer	10

Maximum Transfer Clipping Distance

Syntax

Maximum Transfer Clipping Distance *{=}* *Value*

Summary

Specify the maximum transfer clipping distance. Default will be set to the maximum element length scale of the coupling surface. Any transfer exceeding this clipping distance will throw a fatal error.

Parameter	Value	Default
{=}	{= | are | is}	–
Value	real	1.798e+308

Patch Reconstruction

Syntax

Patch Reconstruction *{=}* *{cubic_bspline | gregory | none}*

Summary

Higher-order reconstruction of patch coordinates

Description

`NONE`

Interpolate coordinate field using standard finite element basis functions

`GREGORY`

Use Gregory patch reconstruction to smooth surface; useful when sending mesh is significantly coarser than receiving mesh. Only applicable to sending the coordinate fields.

`CUBIC_BSPLINE`

Use a cubic B-spline patch reconstruction (which is C^2 continuous) to smooth the surface; useful when sending mesh is significantly coarser than receiving mesh.

Parameter	Value	Default
{=}	{= | are | is}	–
Value	{cubic_bspline | gregory | none}	–

Receive Field

Syntax

Receive Field *{=}* *Value* ([Smooth *Smoothing Iterations* Iterations] | [With Time Interpolation])

Summary

Name of a field to receive from the aero code with optional smoothing or time interpolation.

Parameter	Value	Default
{=}	{= | are | is}	–
Value	string	–
Smoothing Iterations	integer	–

Receive Global

Syntax

Receive Global *{=}* *Value*

Summary

Name of a global variable to receive from the aero code.

Parameter	Value	Default
{=}	{= | are | is}	–
Value	string	–

Send Coordinates Field

Syntax

Send Coordinates Field *{=}* *Value*

Summary

Name of a coordinate vector field to send to the aero code.

Parameter	Value	Default
{=}	{= | are | is}	–
Value	string	–

Send Displacement Field

Syntax

Send Displacement Field *{=}* *Value*

Summary

Name of a displacement vector field to send to the aero code.

Parameter	Value	Default
{=}	{= | are | is}	–
Value	string	–

Send Field

Syntax

Send Field *{=}* *Value*

Summary

Name of a field to send to the aero code.

Parameter	Value	Default
{=}	{= | are | is}	–
Value	string	–

Send Global

Syntax

Send Global *{=}* *Value*

Summary

Name of a global variable to send to the aero code.

Parameter	Value	Default
{=}	{= | are | is}	–
Value	string	–

Symmetry Plane Tolerance

Syntax

Symmetry Plane Tolerance *{=}* *Value*

Summary

Dimensional tolerance for whether a transferred point lies within a symmetry plane

Parameter	Value	Default
{=}	{= | are | is}	–
Value	real	–

Symmetry Planes

Syntax

Symmetry Planes *{=}* *Value. . .*

Summary

List of sidesets that should enforce symmetry properties for B-Spline reconstruction

Parameter	Value	Default
{=}	{= | are | is}	–
Value	string. . .	–

Time Step Mode

Syntax

Time Step Mode *{=}* *{aero_subcycle | implicit | negotiate_minimum | thermal_subcycle}*

Summary

Specify the coupled time stepping behavior.

Description

AERO_SUBCYCLE - Aria chooses its time step and the aero code subcycles until that duration is simulated THERMAL_SUBCYCLE - Aero code chooses its time step and Aria subcycles until that duration is simulated NEGOTIATE_MINIMUM - Both codes compute a candidate time step and advance at the lower of the two IMPLICIT - Both codes negotiate a minimum time step and then iterate the individual app subdomains in a given time step until both apps are converged

Parameter	Value	Default
{=}	{= | are | is}	–
Value	{aero_subcycle | implicit | negotiate_minimum | thermal_subcycle}	–

Transfer Type

Syntax

Transfer Type *{=} {conservative | geometric}*

Summary

Geometric or conservative transfers

Description

This features toggles between an interpolated or geometric based transfer and a conservative scheme that relies on a middle mesh.

`GEOMETRIC`

Use the standard STK transfer which is based on interpolating elemental quantities to desired locations via finite element basis functions

`CONSERVATIVE`

Construct a middle mesh that is a superset of both transfer meshes in order to facilitate a conservative transfer

Parameter	Value	Default
{=}	{= | are | is}	–
Value	{conservative | geometric}	–

Unconverged Implicit Steps

Syntax

Unconverged Implicit Steps *{=} {accept | reject}*

Summary

Specify if unconverged implicit steps should be accepted (default) or rejected.

Parameter	Value	Default
{=}	{= | are | is}	–
Value	{accept | reject}	–

Implicit Tolerances

`begin Implicit Tolerances`

```
  Absolute Norm2 Tolerance For Field {=} Value [With {dof_amax | dof_rms}
↪ Scaling]
```

```
  Default Absolute Norm2 Tolerance {=} Value [With {dof_amax | dof_rms}␣
↪Scaling]

  Default Norm Inf Tolerance {=} Value [With {dof_amax | dof_rms}␣
↪Scaling]

  Default Rms Tolerance {=} Value [With {dof_amax | dof_rms} Scaling]

  Norm Inf Tolerance For Field {=} Value [With {dof_amax | dof_rms}␣
↪Scaling]

  Rms Tolerance For Field {=} Value [With {dof_amax | dof_rms} Scaling]

end Implicit Tolerances
```

Line Commands

Absolute Norm2 Tolerance For

Syntax

Absolute Norm2 Tolerance For *Field {=} Value* [With *{dof_amax | dof_rms}* Scaling]

Summary

Specify the absolute L_2 tolerance for the given field for mpmd coupled solution update delta.

Description

Specify the convergence tolerance for iterating the mpmd subdomains in terms of an absolute nodal L2 norm on the solution field delta. Optionally specify a scaling to normalize delta by before checking against tolerance.

Parameter	Value	Default
Field	string	–
{=}	{= | are | is}	–
Value	real	1.798e+308

Default Absolute Norm2 Tolerance

Syntax

Default Absolute Norm2 Tolerance *{=} Value* [With *{dof_amax | dof_rms}* Scaling]

Summary

Specify the default absolute L_2 tolerance for mpmd coupled solution update delta.

Description

Specify the convergence tolerance for iterating the mpmd subdomains in terms of an absolute nodal L2 norm on the solution field delta. Optionally specify a scaling to normalize delta by before checking against tolerance.

Parameter	Value	Default
{=}	{= | are | is}	–
Value	real	1.798e+308

Default Norm Inf Tolerance

Syntax

Default Norm Inf Tolerance *{=}* *Value* [With *{dof_amax | dof_rms}* Scaling]

Summary

Specify the default L_inf tolerance for mpmd coupled solution update delta.

Description

Specify the convergence tolerance for iterating the mpmd subdomains in terms of an L_Inf norm on the solution field delta. Optionally specify a scaling to normalize delta by before checking against tolerance.

Parameter	Value	Default
{=}	{= | are | is}	–
Value	real	1.798e+308

Default Rms Tolerance

Syntax

Default Rms Tolerance *{=}* *Value* [With *{dof_amax | dof_rms}* Scaling]

Summary

Specify the default RMS tolerance for mpmd coupled solution update delta.

Description

Specify the convergence tolerance for iterating the mpmd subdomains in terms of the RMS on the solution field delta. Optionally specify a scaling to normalize delta by before checking against tolerance.

Parameter	Value	Default
{=}	{= | are | is}	–
Value	real	1.798e+308

Norm Inf Tolerance For

Syntax

Norm Inf Tolerance For *Field* *{=}* *Value* [With *{dof_amax | dof_rms}* Scaling]

Summary

Specify the L_inf tolerance for the given field for mpmd coupled solution update delta.

Description

Specify the convergence tolerance for iterating the mpmd subdomains in terms of an L_Inf norm on the solution field delta. Optionally specify a scaling to normalize delta by before checking against tolerance.

Parameter	Value	Default
Field	string	–
{=}	{= | are | is}	–
Value	real	1.798e+308

Rms Tolerance For

Syntax

Rms Tolerance For *Field {=} Value* [With *{dof_amax | dof_rms}* Scaling]

Summary

Specify the RMS tolerance for the given field for mpmd coupled solution update delta.

Description

Specify the convergence tolerance for iterating the mpmd subdomains in terms of the RMS on the solution field delta. Optionally specify a scaling to normalize delta by before checking against tolerance.

Parameter	Value	Default
Field	string	–
{=}	{= | are | is}	–
Value	real	1.798e+308

Transfer

```
begin Transfer Value

   Coupling Surfaces {=} Value...

   Implicit Receive Field Relaxation {=} Value

   Implicit Send Field Relaxation {=} Value

   Maximum Transfer Clipping Distance {=} Value
```

```
   Patch Reconstruction {=} {cubic_bspline | gregory | none}

   Receive Field {=} Value ([Smooth Smoothing Iterations Iterations]  |␣
↪[With Time Interpolation])

   Receive Global {=} Value

   Send Coordinates Field {=} Value

   Send Displacement Field {=} Value

   Send Field {=} Value

   Send Global {=} Value

   Symmetry Plane Tolerance {=} Value

   Symmetry Planes {=} Value...

   Transfer Type {=} {conservative | geometric}

end Transfer Value
```

Line Commands

Coupling Surfaces

Syntax

Coupling Surfaces *{=} Value...*

Summary

Select the surfaces that are coupled to the aero code.

Parameter	Value	Default
{=}	{= | are | is}	–
Value	string...	–

Implicit Receive Field Relaxation

Syntax

Implicit Receive Field Relaxation *{=} Value*

Summary

Add relaxation to fields received from coupled app for implicit iterations 2+
$f_{k+1} = \omega f'_{k+1} + (1-\omega) f_k$

Parameter	Value	Default
{=}	{= | are | is}	–
Value	real	1

Implicit Send Field Relaxation

Syntax

Implicit Send Field Relaxation *{=}* *Value*

Summary

Add relaxation to fields sent to coupled app for implicit iterations 2+
$f_{k+1} = \omega f'_{k+1} + (1-\omega) f_k$

Parameter	Value	Default
{=}	{= | are | is}	–
Value	real	1

Maximum Transfer Clipping Distance

Syntax

Maximum Transfer Clipping Distance *{=}* *Value*

Summary

Specify the maximum transfer clipping distance. Default will be set to the maximum element length scale of the coupling surface. Any transfer exceeding this clipping distance will throw a fatal error.

Parameter	Value	Default
{=}	{= | are | is}	–
Value	real	1.798e+308

Patch Reconstruction

Syntax

Patch Reconstruction *{=}* *{cubic_bspline | gregory | none}*

Summary

Higher-order reconstruction of patch coordinates

Description

`NONE`

Interpolate coordinate field using standard finite element basis functions

`GREGORY`

Use Gregory patch reconstruction to smooth surface; useful when sending mesh is significantly coarser than receiving mesh. Only applicable to sending the coordinate fields.

`CUBIC_BSPLINE`

Use a cubic B-spline patch reconstruction (which is C^2 continuous) to smooth the surface; useful when sending mesh is significantly coarser than receiving mesh.

Parameter	Value	Default
{=}	{= | are | is}	–
Value	{cubic_bspline | gregory | none}	–

Receive Field

Syntax

Receive Field *{=}* *Value* ([Smooth *Smoothing Iterations* Iterations] | [With Time Interpolation])

Summary

Name of a field to receive from the aero code with optional smoothing or time interpolation.

Parameter	Value	Default
{=}	{= | are | is}	–
Value	string	–
Smoothing Iterations	integer	–

Receive Global

Syntax

Receive Global *{=}* *Value*

Summary

Name of a global variable to receive from the aero code.

Parameter	Value	Default
{=}	{= | are | is}	–
Value	string	–

Send Coordinates Field

Syntax

Send Coordinates Field *{=}* *Value*

Summary

Name of a coordinate vector field to send to the aero code.

Parameter	Value	Default
{=}	{= | are | is}	–
Value	string	–

Send Displacement Field

Syntax

Send Displacement Field *{=}* *Value*

Summary

Name of a displacement vector field to send to the aero code.

Parameter	Value	Default
{=}	{= | are | is}	–
Value	string	–

Send Field

Syntax

Send Field *{=}* *Value*

Summary

Name of a field to send to the aero code.

Parameter	Value	Default
{=}	{= | are | is}	–
Value	string	–

Send Global

Syntax

Send Global *{=}* *Value*

Summary

Name of a global variable to send to the aero code.

Parameter	Value	Default
{=}	{= | are | is}	–
Value	string	–

Symmetry Plane Tolerance

Syntax

Symmetry Plane Tolerance *{=}* *Value*

Summary

Dimensional tolerance for whether a transferred point lies within a symmetry plane

Parameter	Value	Default
{=}	{= | are | is}	–
Value	real	–

Symmetry Planes

Syntax

Symmetry Planes *{=}* *Value. . .*

Summary

List of sidesets that should enforce symmetry properties for B-Spline reconstruction

Parameter	Value	Default
{=}	{= | are | is}	–
Value	string. . .	–

Transfer Type

Syntax

Transfer Type *{=}* *{conservative | geometric}*

Summary

Geometric or conservative transfers

Description

This features toggles between an interpolated or geometric based transfer and a conservative scheme that relies on a middle mesh.

`GEOMETRIC`

Use the standard STK transfer which is based on interpolating elemental quantities to desired locations via finite element basis functions

`CONSERVATIVE`

Construct a middle mesh that is a superset of both transfer meshes in order to facilitate a conservative transfer

Parameter	Value	Default
{=}	{= | are | is}	–
Value	{conservative | geometric}	–

Radiation Coupling

Summary

Specification of parameters for MPMD radiation

```
begin Radiation Coupling Value

   Iterations Per Radiation Solve {=} Value

   Time Steps Per Radiation Solve {=} Value

end Radiation Coupling Value
```

Line Commands

Iterations Per Radiation Solve

Syntax

Iterations Per Radiation Solve *{=}* *Value*

Summary

Set the number of non-linear iterations between each radiation solve

Parameter	Value	Default
{=}	{= | are | is}	–
Value	integer	–

Time Steps Per Radiation Solve

Syntax

Time Steps Per Radiation Solve *{=}* *Value*

Summary

Set the number of time steps between each radiation solve

Parameter	Value	Default
{=}	{= | are | is}	–
Value	integer	–

7.33 Nonlinear Solve Settings

Accept Solution After Maximum Nonlinear Iterations

Syntax

Accept Solution After Maximum Nonlinear Iterations *{=} {false | no | off | on | true | yes}*

Summary

Specifies whether solution is considered to be converged after MAXIMUM NONLINEAR ITERATIONS even if none of the convergence criteria are satisfied.

Parameter	Value	Default
{=}	{= | are | is}	–
Bool	{false | no | off | on | true | yes}	FALSE

Filter Nonlinear Solution

Syntax

Filter Nonlinear Solution For *AssociatedDoF FilterType {=} Values. . .*

Summary

Restrict the value of a solution variable used in the nonlinear solver iterations of a solution step.

Description

The nonlinear solution can be restricted to a range with upper and lower limits using the FILTER_TYPE = RANGE while supplying two bounding values. To set an upper or lower bound use FILTER_TYPE = MAXIMUM or FILTER_TYPE = MINIMUM while supplying a single bounding value.

Parameter	Value	Default
AssociatedDoF	string	–
FilterType	string	–
{=}	{= | are | is}	–
Values	string. . .	–

Maximum Accepted Nonlinear Correction

Syntax

Maximum Accepted Nonlinear Correction *{=}* *mnlc*

Summary

Maximum allowed nonlinear correction tolerance during residual acceptance.

Parameter	Value	Default
{=}	{= | are | is}	–
mnlc	real	1e16

Maximum Accepted Nonlinear Residual

Syntax

Maximum Accepted Nonlinear Residual *{=}* *mnlr*

Summary

Maximum allowed nonlinear residual tolerance during nonlinear correction acceptance.

Parameter	Value	Default
{=}	{= | are | is}	–
mnlr	real	1e16

Maximum Line Search Steps

Syntax

Maximum Line Search Steps *{=}* *MaxSteps*

Summary

Maximum number of line search steps per nonlinear iteration.

Description

Only applicable when the LINE_SEARCH nonlinear solution strategy is used.

Parameter	Value	Default
{=}	{= | are | is}	–
MaxSteps	integer	10

Maximum Nonlinear Iterations

Syntax

Maximum Nonlinear Iterations *{=}* *Np*

Summary

Number of allowable nonlinear iterations in one linear solution step.

Description

The solution step will terminate when the number of nonlinear iterations are exceeded. Default value = 20.

Parameter	Value	Default
{=}	{= | are | is}	–
Np	integer	20

Minimum Nonlinear Solves

Syntax

Minimum Nonlinear Solves *{=}* *Np*

Summary

Minimum number of actual nonlinear solves that are required.

Description

Each nonlinear iteration involves first assembling the nonlinear residual and Jacobian matrix and second solving the linear system for a nonlinear correction that is applied to the estimate of the nonlinear solution.

The nonlinear solver will continue iterating until this minimum number of solves are performed. The default value is one (1) but setting this to zero can be useful in some situations.

This command is identical to “minimum nonlinear iterations = X”.

Parameter	Value	Default
{=}	{= | are | is}	–
Np	integer	1

Nonlinear Correction Ratio Tolerance

Syntax

Nonlinear Correction Ratio Tolerance *{=}* *Pt*

Summary

Convergence tolerance for the nonlinear correction. It is satisfied if the nonlinear correction is less than this specified multiple of the initial nonlinear correction.

Parameter	Value	Default
{=}	{= | are | is}	–
Pt	real	–

Nonlinear Correction Scaling

Syntax

Nonlinear Correction Scaling *{=}* *{dof_amax | none}*

Summary

Specifies how to scale the nonlinear correction norm.

Description

NONE (default) - The unscaled nonlinear corrections will be used.

DOF_AMAX (recommended) - The corrections for each equation will be scaled by max(1., abs(max(solution_N)))

Parameter	Value	Default
{=}	{= | are | is}	–
Strategy	{dof_amax | none}	NONE

Nonlinear Correction Tolerance

Syntax

Nonlinear Correction Tolerance *{=}* *Pt*

Summary

Convergence tolerance of nonlinear correction norm for the iteration. Satisfaction of this criterion is sufficient for the iteration to be considered finished.

Parameter	Value	Default
{=}	{= | are | is}	–
Pt	real	1.0e-6

Nonlinear Relaxation Factor

Syntax

Nonlinear Relaxation Factor *{=}* *Prf*

Summary

Weighting factor for fraction of a new nonlinear solution that will be applied to the linear solution update. Weighting factor must lie in the range 0.0 to 1.0.

Functions of iteration count (i) are also supported. The function string must be enclosed in quotes if it has spaces or commas.

Parameter	Value	Default
{=}	{= | are | is}	–
Prf	"string"	1.0

Nonlinear Residual Minimum Convergence Rate

Syntax

Nonlinear Residual Minimum Convergence Rate *{=}* *MinRate* [Number Of Steps *{=}* *MinSteps*]

Summary

Set a minimum convergence rate based on the nonlinear residual At each nonlinear iteration the condition $resid_N / resid_{N-MinSteps} < MinRate^{MinSteps}$. If that condition is not met then the time step fails and is retried with a shorter dt. The usage of this test is modified if the command "USE SECONDARY LINEAR SOLVER" is specified. See the description of that command for details.

Parameter	Value	Default
{=}	{= | are | is}	–
MinRate	real	–

Nonlinear Residual Ratio Tolerance

Syntax

Nonlinear Residual Ratio Tolerance *{=}* *Pt*

Summary

Convergence tolerance for the ratio of the nonlinear residual to the initial nonlinear residual for the solution iteration. Convergence is satisfied if the ratio is less than this specified multiple of the initial nonlinear residual.

Parameter	Value	Default
{=}	{= | are | is}	–
Pt	real	0.0

Nonlinear Residual Scaling

Syntax

Nonlinear Residual Scaling *{=} {custom | dof_amax | non_dimensional | none}*

Summary

Specifies how to scale the nonlinear residuals.

Description

NONE - The unscaled nonlinear residuals will be used.

DOF_AMAX (recommended) - The residuals for each equation will be scaled by `max(1., abs(max(solution_N)))`

CUSTOM - Provide scales for individual DoFs (default 1)

NON_DIMENSIONAL - This option scales the nonlinear residual by the inverse of `W = D * dX_scale` resulting in a non-dimensional residual, where `D = diag(J)`, is the diagonal of the current Jacobian and `dX_scale` is a matrix whose diagonal is the DOF_AMAX residual scaling vector. Note that scaling the residual implies left scaling of the linear system (i.e., matrix rows are scaled as well). If a zero diagonal entry is encountered, the one norm of the matrix rows is used instead.

Parameter	Value	Default
{=}	{= | are | is}	–
Strategy	{custom | dof_amax | non_dimensional | none}	NONE

Nonlinear Residual Tolerance

Syntax

Nonlinear Residual Tolerance *{=}* *Pt*

Summary

Convergence tolerance of nonlinear residual for the iteration. Satisfaction of this criterion is sufficient for the iteration to be considered finished.

Parameter	Value	Default
{=}	{= | are | is}	–
Pt	real	1.0e-6

Nonlinear Solution Strategy

Syntax

Nonlinear Solution Strategy *{=} {line_search | newton | newton_finite_difference | newton_lagged}* [*FormulationType*]

Summary

Specifies the nonlinear solution strategy.

Description

There is no default value so this line command is required.

Parameter	Value	Default
{=}	{= | are | is}	–
Strategy	{line_search | newton | newton_finite_difference | newton_lagged}	–

Use Dof Averaged Nonlinear Residual

Syntax

Use Dof Averaged Nonlinear Residual

Summary

Select whether to use a nonlinear residual normalized by the number of degrees of freedom instead of the raw global residual reported by the linear system.

Description

Specifies that the reported nonlinear residual is scaled (divided by) the number of DOF in the problem. In selecting appropriate values for the NONLINEAR RESIDUAL TOLERANCE users are cautioned that the average nonlinear residual reported values may be several orders of magnitude smaller than when using the global nonlinear residual. This option is provided to allow comparison with residual values from other applications.

Use Linear Solver

Syntax

Use Linear Solver *Solver* [*SolverName*]

Summary

Specifies which solver to use with this region, optionally providing a name for the solver (default: the_linear_solver)

Parameter	Value	Default
Solver	string	–

Use Secondary Linear Solver

Syntax

Use Secondary Linear Solver *Solver...* [For *NumSteps...* Steps]

Summary

Specifies which auxiliary solver(s) to use with this region. Every nonlinear solve will begin employing either the primary solver specified by the command "USE LINEAR SOLVER", of one of the auxiliary solver(s) specified by this command. A new auxiliary solver will take over for the current solver in two cases. First, if the current linear solver fails due to a critical numerical issue (e.g. solver breakdown), the time step is failed and the next linear solver is used. Second, if the current linear solver fails due to a loss of accuracy (-3) or does not converge in the maximum iteration count, the nonlinear residual is observed. If the nonlinear residual fails to decrease, the time step is failed and the next linear solver is used. Note that, in both cases, the time step is re-tried at the same time step size. This process is repeated until all auxiliary solvers have been tried. If none of the auxiliary solvers work, the time step will fail with the usual new time step size selection. This command allows for successively more robust solvers to be used if the linear solver struggles to converge at some point in a simulation. This only functions if both the primary and auxiliary solver(s)s are Tpetra-based. This should be considered a feature for expert users.

By default, each auxiliary solver is active for 10 time steps once it has been activated. You can change that by adding the optional "FOR X1, X2, . . . , XN STEPS" command suffix, where the step count order corresponds to the solver order specified in this command. After the number of time steps, the previous auxiliary or primary linear solver is restored.

Parameter	Value	Default
Solver	string...	–

7.34 Skinning

Skinned Surface

Scope

Aria Region

Summary

Creates a skinned surface from blocks with the ability to specify subsets of surface, blocks, etc.

`begin Skinned Surface` *Name*

```
   Excluding Blocks {=} value...

   Excluding Surfaces {=} value...

   Internal Faces {=} {exclude | include}

   Intersecting Blocks {=} value...

   Intersecting Surfaces {=} value...

   Touching Blocks {=} value...

   Touching Surfaces {=} value...

end Skinned Surface Name
```

Line Commands

Excluding Blocks

Syntax

Excluding Blocks *{=} value...*

Summary

Select the union of blocks to exclude when creating sideset.

Parameter	Value	Default
{=}	{= | are | is}	–
value	string...	–

Excluding Surfaces

Syntax

Excluding Surfaces *{=} value...*

Summary

Select the union of surfaces to exclude when creating sideset.

Parameter	Value	Default
{=}	{= | are | is}	–
value	string...	–

Internal Faces

Syntax

Internal Faces *{=} {exclude | include}*

Summary

Controls if internal faces (those not on the exposed boundary of the domain) are included in the surface. The default is to exclude them UNLESS intersections are requested.

Parameter	Value	Default
{=}	{= | are | is}	–
value	{exclude | include}	–

Intersecting Blocks

Syntax

Intersecting Blocks *{=} value. . .*

Summary

Select the intersection of blocks to include when creating sideset.

Parameter	Value	Default
{=}	{= | are | is}	–
value	string. . .	–

Intersecting Surfaces

Syntax

Intersecting Surfaces *{=} value. . .*

Summary

Select the intersection of surfaces to include when creating sideset.

Parameter	Value	Default
{=}	{= | are | is}	–
value	string. . .	–

Touching Blocks

Syntax

Touching Blocks *{=} value. . .*

Summary

Select the union of blocks to include when creating sideset.

Parameter	Value	Default
{=}	{= | are | is}	–
value	string. . .	–

Touching Surfaces

Syntax

Touching Surfaces *{=}* *value...*

Summary

Select the union of surfaces to include when creating sideset.

Parameter	Value	Default
{=}	{= | are | is}	–
value	string...	–

Skinned Shell

Scope

Aria Region

Summary

Creates a skinned shell from blocks with the ability to specify subsets of surface, blocks, etc. Note that while this is similar to created a skinned surface, this block creates new shell elements as opposed to collecting faces of existing elements

```
begin Skinned Shell Name

   Excluding Blocks {=} value...

   Excluding Surfaces {=} value...

   Interpolate {mesh | solid} Displacements From part...

   Node Merge Tolerance {=} value

   Touching Blocks {=} value...

   Touching Surfaces {=} value...

   Use Material value

end Skinned Shell Name
```

Line Commands

Excluding Blocks

Syntax

Excluding Blocks *{=} value. . .*

Summary

Select the union of blocks to exclude when creating shell elements.

Parameter	Value	Default
{=}	{= | are | is}	–
value	string. . .	–

Excluding Surfaces

Syntax

Excluding Surfaces *{=} value. . .*

Summary

Select the union of surfaces to exclude when creating shell elements.

Parameter	Value	Default
{=}	{= | are | is}	–
value	string. . .	–

Interpolate

Syntax

Interpolate *{mesh | solid}* Displacements From *part. . .*

Summary

If this option is specified, displacement of the shell is found by projecting its nodes onto the specified set of parts. The resulting displacement is used to populate the MESH/SOLID displacement solution. Note that an XFER equation will be declared on the shell part in order to do this if the eq is not present.

Parameter	Value	Default
value	{mesh | solid}	–
part	string. . .	–

Node Merge Tolerance

Syntax

Node Merge Tolerance *{=} value*

Summary

Specify the relative tolerance (scaled by characteristic elem length scale) for distance to merge nodes.

Parameter	Value	Default
{=}	{= | are | is}	–
value	real	–

Touching Blocks

Syntax

Touching Blocks *{=}* *value. . .*

Summary

Select the union of blocks to include when creating shell elements.

Parameter	Value	Default
{=}	{= | are | is}	–
value	string. . .	–

Touching Surfaces

Syntax

Touching Surfaces *{=}* *value. . .*

Summary

Select the union of surfaces to include when creating shell elements.

Parameter	Value	Default
{=}	{= | are | is}	–
value	string. . .	–

Use Material

Syntax

Use Material *value*

Summary

Select the material to apply.

Parameter	Value	Default
value	string	–

7.35 Segregated Solve Settings

Segregated Global Minimum Convergence Rate

Syntax

Segregated Global Minimum Convergence Rate *{=}* *MinRate* [Number Of Steps *{=}* *MinSteps*]

Summary

Set a minimum convergence rate based on the initial nonlinear residual of each equation system in the problem. At each segregated iteration the equation system with the largest initial residual is tested using

$$r_N / r_{N-\text{MinSteps}} < \text{MinRate}^{\text{MinSteps}}$$

If this condition is not met then the time step fails and is retried with a shorter dt.

Parameter	Value	Default
{=}	{= | are | is}	–
MinRate	real	–

Segregated Global Nonlinear Initial Residual Tolerance

Syntax

Segregated Global Nonlinear Initial Residual Tolerance *{=}* *Pt*

Summary

Convergence tolerance for global segregated nonlinear residual for the iteration. The iteration is considered finished when all of the individual equation systems have an initial nonlinear residual less than the given tolerance. This is cheaper than other tests that require recomputing the nonlinear residual. However, care should be taken since the initial nonlinear residual of the individual equation systems might not be a good measure of the final nonlinear residual of those systems due to deltas produced by other equation systems.

Parameter	Value	Default
{=}	{= | are | is}	–
Pt	real	–

Segregated Global Nonlinear Number Of Steps

Syntax

Segregated Global Nonlinear Number Of Steps *{=}* *Number_of_steps* [Fail Step *{=}* *{false | no | off | on | true | yes}*]

Summary

Maximum number of segregated solution steps within a time step. Depending on the fail value the solution will either be accepted, or the time step will fail when the maximum number of steps is reached.

Parameter	Value	Default
{=}	{= | are | is}	–
Number_of_steps	integer	–

Segregated Global Nonlinear Relative Residual Tolerance

Syntax

Segregated Global Nonlinear Relative Residual Tolerance *{=}* *Pt*

Summary

Convergence tolerance for global segregated nonlinear residual for the iteration. Satisfaction of this criterion is sufficient for the iteration to be considered finished. This line command can be used when the user desires to enforce a global relative residual tolerance for the entire segregated system that may be different from the individual equation systems value.

Parameter	Value	Default
{=}	{= | are | is}	–
Pt	real	–

Segregated Global Nonlinear Residual Tolerance

Syntax

Segregated Global Nonlinear Residual Tolerance *{=}* *Pt* [Minimum Steps *{=}* *minStepsValue*]

Summary

Convergence tolerance for global segregated nonlinear residual for the iteration. Satisfaction of this criterion is sufficient for the iteration to be considered finished. This line command can be used when the user desires to enforce a global residual enforcement for the entire segregated system that may be different from the individual equation systems value

Parameter	Value	Default
{=}	{= | are | is}	–
Pt	real	–

Segregated Global Nonlinear Target Number Of Steps Per Time Step

Syntax

Segregated Global Nonlinear Target Number Of Steps Per Time Step *{=}* *target_iterations*

Summary

Set a target number of segregated iterations per time step to use for time step control. If more than the specified number of iterations are required to converge a time step the next time step size will be decreased proportional to the ratio of the target number and the number taken. If a maximum number of allowed segregated iterations is set it is recommended to set the target to 1-2 fewer iterations to avoid an excessive number of failed time steps.

Parameter	Value	Default
{=}	{= | are | is}	–
target_iterations	integer	–

7.36 Output

Results Output

Scope

Aria Region, Input_Output Region

Summary

Describes the location and type of the output stream used for outputting results for the enclosing region.

```
begin Results Output Label

   Additional Steps {=} List_of_steps...

   Additional Times {=} List_of_times...

   At Step n {increment | interval} {=} m

   At Time Dt1 {increment | interval} {=} Dt2
```

```
  Auto Output {all | element | global | nodal} User Defined Variables [␣
↪In UserOutputResultsList...  ]

  Auto Output {all | element | global | nodal} Variables

  Component Separator Character {=} Separator

  Database Name {=} StreamName

  Database Type {=} {catalyst | catalyst_exodus | cgns | dof | dof_
↪exodus | exodus | exodusii | exonull | generated | genesis | null |␣
↪parallel_exodus | textmesh}

  Edge VariableList...

  Element VariableList...

  Enable Large Ids

  Exclude {=} [ ElementBlockList...  ]

  Exists {=} {abort | add_suffix | append | overwrite}

  Face VariableList...

  Flush Interval {=} Option

  Global VariableList...

  Include {=} [ ElementBlockList...  ]

  Nodal VariableList...

  Node VariableList...

  Nodeset VariableList...

  Output Mesh {=} {exposed surface | refined | unrefined}

  Output On Signal {=} {sigabrt | sigalrm | sigfpe | sighup | sigill |␣
↪sigint | sigkill | sigpipe | sigquit | sigsegv | sigterm | sigusr1 |␣
↪sigusr2}

  Overwrite {=} {false | no | off | on | true | yes}
```

```
   Property PropertyName {=} PropertyValue

   Sideset VariableList...

   Start Time {=} Start_time

   Surface VariableList...

   Synchronize Output

   Termination Time {=} Final_time

   Timeseries Name {=} filename

   Timestep Adjustment Interval {=} Nsteps

   Title

   Use Dynamic Topology Io

   Use External Control

   Use Output Scheduler Timer_name

   begin Catalyst Label
   end

end Results Output Label
```

Line Commands

Additional Steps

Syntax

Additional Steps *{=}* *List_of_steps...*

Summary

Additional simulation steps when output should occur.

Parameter	Value	Default
{=}	{= | are | is}	–
List_of_steps	integer...	–

Additional Times

Syntax

Additional Times *{=} List_of_times. . .*

Summary

Additional simulation times when output should occur.

Parameter	Value	Default
{=}	{= | are | is}	–
List_of_times	real. . .	–

At Step

Syntax

At Step *n {increment | interval} {=} m*

Summary

Specify an output interval in terms of the internal iteration step count. The first step specifies the step count at the beginning of this interval and the second step specifies the output frequency to be used within this interval.

Parameter	Value	Default
n	integer	–
Option	{increment | interval}	–
{=}	{= | are | is}	–
m	integer	–

At Time

Syntax

At Time *Dt1 {increment | interval} {=} Dt2*

Summary

Specify an output interval in terms of the internal simulation time. The first time specifies the time at the beginning of this time interval and the second time specifies the output frequency to be used within this interval.

Parameter	Value	Default
Dt1	real	–
Option	{increment | interval}	–
{=}	{= | are | is}	–
Dt2	real	–

Auto Output

Syntax

Auto Output *{all | element | global | nodal}* User Defined Variables [In *UserOutputResultsList. . .*]

Summary

Allows users to automatically output all user output defined variables for the type requested.

Parameter	Value	Default
auto_output_type_3	{all | element | global | nodal}	–

Auto Output

Syntax

Auto Output *{all | element | global | nodal}* Variables

Summary

Allows users to automatically output all user output defined variables for the type requested.

Parameter	Value	Default
auto_output_type_3	{all | element | global | nodal}	–

Component Separator Character

Syntax

Component Separator Character *{=} Separator*

Summary

The separator is the single character used to separate the output variable basename (e.g. “stress”) from the suffices (e.g. “xx”, “yy”) when displaying the names of the individual variable components. For example, the default separator is “_”, which results in names similar to “stress_xx”, “stress_yy”, . . . “stress_zx”. To eliminate the separator, specify an empty string (“”) or NONE.

Parameter	Value	Default
{=}	{= | are | is}	–
Separator	string	–

Database Name

Syntax

Database Name *{=} StreamName*

Summary

The base name of the database containing the output results. If the filename begins with the ‘/’ character, it is an absolute path; otherwise, the path to the

current directory will be prepended to the name. If this line is omitted, then a filename will be created from the basename of the input file with a ".e" suffix appended.

Parameter	Value	Default
{=}	{= | are | is}	–
StreamName	string	–

Database Type

Syntax

Database Type *{=} {catalyst | catalyst_exodus | cgns | dof | dof_exodus | exodus | exodusii | exonull | generated | genesis | null | parallel_exodus | textmesh}*

Summary

The database type/format to be used for the output results.

Parameter	Value	Default
{=}	{= | are | is}	–
DatabaseType	{catalyst | catalyst_exodus | cgns | dof | dof_exodus | exodus | exodusii | exonull | generated | genesis | null | parallel_exodus | textmesh}	–

Edge

Syntax

Edge *VariableList. . .*

Summary

Define the variables that should be written to the results database. If "variable" is entered, then its name will be used on the output database. If "variable as db_name" is entered, then "db_name" will be the name used on the database for the internal variable "variable". Multiple "variable" or "variable as db_name" entries are allowed on the same line. The entities that this variable are written to can also be limited or specified with "exclude list_of_entities" or "include list_of_entities". Edge variables are not supported for all database types.

Parameter	Value	Default
VariableList	string. . .	–

Element

Syntax

Element *VariableList. . .*

Summary

Define the variables that should be written to the results database. If “variable” is entered, then its name will be used on the output database. If “variable as db_name” is entered, then “db_name” will be the name used on the database for the internal variable “variable”. Multiple “variable” or “variable as db_name” entries are allowed on the same line. The entities that this variable are written to can also be limited or specified with “exclude list_of_entities” or “include list_of_entities”

Parameter	Value	Default
VariableList	string. . .	–

Enable Large Ids

Syntax

Enable Large Ids

Summary

Enable 64 bit entity IDs for output

Exclude

Syntax

Exclude *{=}* [*ElementBlockList. . .*]

Summary

Specify that the results file will only contain a subset of the element blocks in the analysis model. The element_block_list lists only the blocks which will not be output to the results database.

Parameter	Value	Default
{=}	{= | are | is}	–

Exists

Syntax

Exists *{=} {abort | add_suffix | append | overwrite}*

Summary

Specify the behavior when creating this database and there is an existing file with the same name. The default behavior is “OVERWRITE” which deletes the existing file and creates a new file of the same name. “APPEND” will (if possible) append the new data to the end of the existing file. “ABORT” will print an error message and end the analysis. “ADD_SUFFIX” will add a -s???? suffix where the ???? is replaced by a sequential number starting at 0002.

Parameter	Value	Default
{=}	{= | is}	–
Option2	{abort | add_suffix | append | overwrite}	–

Face

Syntax

Face *VariableList. . .*

Summary

Define the variables that should be written to the results database. If "variable" is entered, then its name will be used on the output database. If "variable as db_name" is entered, then "db_name" will be the name used on the database for the internal variable "variable". Multiple "variable" or "variable as db_name" entries are allowed on the same line. The entities that this variable are written to can also be limited or specified with "exclude list_of_entities" or "include list_of_entities". Face variables are not supported for all database types.

Parameter	Value	Default
VariableList	string. . .	–

Flush Interval

Syntax

Flush Interval *{=}* *Option*

Summary

The minimum time interval (in seconds) at which output will be explicitly flushed to disk. The default is 10 seconds.

Parameter	Value	Default
{=}	{= | are | is}	–
Option	integer	10

Global

Syntax

Global *VariableList. . .*

Summary

Define the global variables that should be written to the results database. If "variable" is entered, then its name will be used on the output database. If "variable as db_name" is entered, then "db_name" will be the name used on the database for the internal variable "variable". Multiple "variable" or "variable as db_name" entries are allowed on the same line.

Parameter	Value	Default
VariableList	string. . .	–

Include

Syntax

Include *{=}* [*ElementBlockList. . .*]

Summary

Specify that the results file will only contain a subset of the element blocks in the analysis model. The element_block_list lists only the blocks which will be output to the results database.

Parameter	Value	Default
{=}	{= | are | is}	–

Nodal

Syntax

Nodal *VariableList. . .*

Summary

Define the nodal variables that should be written to the results database. If "variable" is entered, then its name will be used on the output database. If "variable as db_name" is entered, then "db_name" will be the name used on the database for the internal variable "variable". Multiple "variable" or "variable as db_name" entries are allowed on the same line.

Parameter	Value	Default
VariableList	string. . .	–

Node

Syntax

Node *VariableList. . .*

Summary

Define the nodal variables that should be written to the results database. If "variable" is entered, then its name will be used on the output database. If "variable as db_name" is entered, then "db_name" will be the name used on the database for the internal variable "variable". Multiple "variable" or "variable as db_name" entries are allowed on the same line.

Parameter	Value	Default
VariableList	string. . .	–

Nodeset

Syntax

Nodeset *VariableList. . .*

Summary

Define the variables that should be written to the results database. If "variable" is entered, then its name will be used on the output database. If "variable as db_name" is entered, then "db_name" will be the name used on the database for the internal variable "variable". Multiple "variable" or "variable as db_name" entries are allowed on the same line. The entities that this variable are written to can also be limited or specified with "exclude list_of_entities" or "include list_of_entities". Nodeset variables are not supported for all database types.

Parameter	Value	Default
VariableList	string. . .	–

Output Mesh

Syntax

Output Mesh *{=} {exposed surface | refined | unrefined}*

Summary

Use this command to turn on "unrefined" as the output mesh. The default behavior is "refined", in which field variables are output on the current mesh, which may have been refined (either uniformly or adaptively) or had its topology altered in some way (e.g., dynamic load balancing) with respect to the original mesh read from the input file. By specifying "Output Mesh = unrefined", all output variables are output only on the original mesh objects read from the input file.

Parameter	Value	Default
{=}	{= | are | is}	–
OutputMesh	{exposed surface | refined | unrefined}	–

Output On Signal

Syntax

Output On Signal *{=} {sigabrt | sigalrm | sigfpe | sighup | sigill | sigint | sigkill | sigpipe | sigquit | sigsegv | sigterm | sigusr1 | sigusr2}*

Summary

When the specified signal is raised, the output stream associated with this block will be output.

Parameter	Value	Default
{=}	{= | are | is}	–
Signals	{sigabrt | sigalrm | sigfpe | sighup | sigill | sigint | sigkill | sigpipe | sigquit | sigsegv | sigterm | sigusr1 | sigusr2}	–

Overwrite

Syntax

Overwrite *{=} {false | no | off | on | true | yes}*

Summary

(DEPRECATED, Use EXISTS) Specify whether the database should be overwritten if it exists. The default behavior is to overwrite unless this command is specified in the output block and either off, false, or no is specified.

Parameter	Value	Default
{=}	{= | is}	–
Option2	{false | no | off | on | true | yes}	–

Property

Syntax

Property *PropertyName {=} PropertyValue*

Summary

Define a database property named “PropertyName” with the value “PropertyValue”. If PropertyValue consists of all digits, it will define an integer property. If PropertyValue is “true” or “yes” or “false” or “no”, it will define a logical property; otherwise it will define a string property. Supported properties are typically database dependent; Current properties are:

- COMPRESSION_LEVEL = [0..9] (off)
- COMPRESSION_SHUFFLE = true|false|on|off (off)
- FILE_TYPE = netcdf4 (forces use of netcdf-4 hdf5-based file) (netcdf3)
- INTEGER_SIZE_DB = 4|8 (4)
- INTEGER_SIZE_API = 4|8 (4)
- REAL_SIZE_DB = 4|8 (8 is default)
- LOGGING = true|false|on|off (off)
- MAX_NAME_LENGTH = value (32)

Parameter	Value	Default
PropertyName	string	–
{=}	{= | are | is}	–
PropertyValue	string	–

Sideset

Syntax

Sideset *VariableList. . .*

Summary

Define the variables that should be written to the results database. If “variable” is entered, then its name will be used on the output database. If “variable as db_name” is entered, then “db_name” will be the name used on the database for the internal variable “variable”. Multiple “variable” or “variable as db_name” entries are allowed on the same line. The entities that this variable are written to can also be limited or specified with “exclude list_of_entities” or “include list_of_entities”. Face variables are not supported for all database types.

Parameter	Value	Default
VariableList	string. . .	–

Start Time

Syntax

Start Time *{=}* *Start_time*

Summary

Specify the time to start outputting results from this output request block. This time overrides all ‘at time’ and ‘at step’ specifications.

Parameter	Value	Default
{=}	{= | are | is}	–
Start_time	real	–

Surface

Syntax

Surface *VariableList. . .*

Summary

Define the variables that should be written to the results database. If “variable” is entered, then its name will be used on the output database. If “variable as db_name” is entered, then “db_name” will be the name used on the database for the internal variable “variable”. Multiple “variable” or “variable as db_name” entries are allowed on the same line. The entities that this variable are written to

can also be limited or specified with “exclude list_of_entities” or “include list_of_entities”. Face variables are not supported for all database types.

Parameter	Value	Default
VariableList	string...	–

Synchronize Output

Syntax

Synchronize Output

Summary

In an analysis with multiple regions, it is sometimes desirable to synchronize the output of results data between the regions. This can be done by adding the SYNCHRONIZE OUTPUT command line to the results output block. If a results block has this set, then it will write output whenever a previous region writes output. The ordering of regions is based on the order in the input file, algorithmic considerations, or by solution control specifications.

Although the USE OUTPUT SCHEDULER command line can also synchronize output between regions, the SYNCHRONIZE OUTPUT command line will synchronize the output with regions where the output frequency is not under the direct control of the Sierra IO system. Examples of this are typically coupled applications where one or more of the codes are not Sierra-based applications such as Alegra and CTH. A results block with SYNCHRONIZE OUTPUT specified will also synchronize its output with the output of the external code.

The SYNCHRONIZE OUTPUT command can be used with other output scheduling commands such as time-based or step-based output specifications.

Termination Time

Syntax

Termination Time *{=}* *Final_time*

Summary

Specify the time to stop outputting results from this output request block.

Parameter	Value	Default
{=}	{= | are | is}	–
Final_time	real	–

Timeseries Name

Syntax

Timeseries Name *{=}* *filename*

Summary

Optionally specify a filename for a timeseries file that outputs the root database filename in the order that they are written. This is useful when running on large numbers of processors with many mesh-mods that cause simple disk operations to hang.

Parameter	Value	Default
{=}	{= | are | is}	–
filename	string	–

Timestep Adjustment Interval

Syntax

Timestep Adjustment Interval *{=}* *Nsteps*

Summary

Specify the number of steps to 'look ahead' and adjust the timestep to ensure that the specified output times or simulation end time will be hit 'exactly'.

Parameter	Value	Default
{=}	{= | are | is}	–
Nsteps	integer	–

Title

Syntax

Title

Summary

Specify the title to be used for this specific output block.

Use Dynamic Topology Io

Syntax

Use Dynamic Topology Io

Summary

Specify that the app use IO for dynamic topology modifications where the output files are stored in a single database. Legacy file format for dynamically changing topology results in the creation of multiple files for each output on a mesh modification. This option leverages the ability of netCDF to create mesh groups within a single database and concatenate all mesh files into one. The names of each mesh group are of the form IOSS_MESH_GROUP-??? where ??? is the 1-based output index 1, 2, . . . , 10, ..., 100, . . . Please note that netCDF has a current limit of 65,536 groups

Use External Control

Syntax

Use External Control

Summary

Do not schedule any results output but instead depend on an externally coupled application to explicitly call for results output.

Use Output Scheduler

Syntax

Use Output Scheduler *Timer_name*

Summary

Associates a predefined output scheduler with this output block (results, restart, heartbeat, or history).

Parameter	Value	Default
Timer_name	string	–

Heartbeat

Scope

Aria Region, Input_Output Region

Summary

Describes the location and type of the output stream used for outputting the heartbeat information for the enclosing region.

```
begin Heartbeat Label

   Additional Steps {=} List_of_steps...

   Additional Times {=} List_of_times...

   Append {=} {false | off | on | true}

   At Step n {increment | interval} {=} m

   At Time Dt1 {increment | interval} {=} Dt2

   Auto Output {all | element | global | nodal} User Defined Variables [␣
↪In UserOutputHeartBeatList...  ]

   Element [ VariableList...  ]

   Exists {=} {abort | append | overwrite}
```

```
  Face [ VariableList...  ]

  Format {=} {csv | original | spyhis}

  Global [ Variables...  ]

  Labels {=} {off | on}

  Legend {=} {off | on}

  Monitor {= | the} {history | restart | results}

  Nodal [ VariableList...  ]

  Node [ VariableList...  ]

  Nodeset [ VariableList...  ]

  Output On Signal {=} {sigabrt | sigalrm | sigfpe | sighup | sigill |␣
↪sigint | sigkill | sigpipe | sigquit | sigsegv | sigterm | sigusr1 |␣
↪sigusr2}

  Precision {=} Precision

  Start Time {=} Start_time

  Stream Name {=} OutputFilename

  Synchronize Output

  Termination Time {=} Final_time

  Timestamp Format

  Timestep Adjustment Interval {=} Nsteps

  Use Output Scheduler Timer_name

  Variable {=} {edge | element | face | global | nodal | node} Variable_
↪list...

end Heartbeat Label
```

Line Commands

Additional Steps

Syntax

Additional Steps *{=} List_of_steps. . .*

Summary

Additional simulation steps when output should occur.

Parameter	Value	Default
{=}	{= | are | is}	–
List_of_steps	integer. . .	–

Additional Times

Syntax

Additional Times *{=} List_of_times. . .*

Summary

Additional simulation times when output should occur.

Parameter	Value	Default
{=}	{= | are | is}	–
List_of_times	real. . .	–

Append

Syntax

Append *{=} {false | off | on | true}*

Summary

Specifies whether the heartbeat file is appended if it exists. By default, the file is appended if restart is requested and not if restart is not requested. This option does not work for automatic restarts because a new heartbeat file is written with each auto restart.

Parameter	Value	Default
{=}	{= | are | is}	–
Option	{false | off | on | true}	–

At Step

Syntax

At Step *n {increment | interval} {=} m*

Summary

Specify an output interval in terms of the internal iteration step count. The first step specifies the step count at the beginning of this interval and the second step specifies the output frequency to be used within this interval.

Parameter	Value	Default
n	integer	–
Option	{increment | interval}	–
{=}	{= | are | is}	–
m	integer	–

At Time

Syntax

At Time *Dt1 {increment | interval} {=} Dt2*

Summary

Specify an output interval in terms of the internal simulation time. The first time specifies the time at the beginning of this time interval and the second time specifies the output frequency to be used within this interval.

Parameter	Value	Default
Dt1	real	–
Option	{increment | interval}	–
{=}	{= | are | is}	–
Dt2	real	–

Auto Output

Syntax

Auto Output *{all | element | global | nodal}* User Defined Variables [In *UserOutputHeartBeatList. . .*]

Summary

Allows users to automatically output all user output defined variables for the type requested.

Parameter	Value	Default
auto_output_type_4	{all | element | global | nodal}	–

Element

Syntax

Element [*VariableList. . .*]

Summary

Define the element variables that should be written to the heartbeat database. The syntax is: “element {internal_name} at element {id} as {DBname}” or “element {internal_name} nearest location X, Y, Z as {DBname}”.

Where {internal_name} is the name of the variable in the Sierra application; and {DBname} is the name as it should appear on the heartbeat database.

Exists

Syntax

Exists *{=} {abort | append | overwrite}*

Summary

Specify the behavior when creating this database and there is an existing file with the same name. The default behavior is “OVERWRITE” which deletes the existing file and creates a new file of the same name. “APPEND” will (if possible) append the new data to the end of the existing file. “ABORT” will print an error message and end the analysis.

Parameter	Value	Default
{=}	{= | is}	–
Option2	{abort | append | overwrite}	–

Face

Syntax

Face [*VariableList. . .*]

Summary

Define the face variables that should be written to the heartbeat database. The syntax is: “face {internal_name} at face {id} as {DBname}” or “face {internal_name} nearest location X, Y, Z as {DBname}”.

Where {internal_name} is the name of the variable in the Sierra application; and {DBname} is the name as it should appear on the heartbeat database.

Format

Syntax

Format *{=} {csv | original | spyhis}*

Summary

The stream type/format to be used for the output results.The only three options at this time are ‘Original’ which is the old default Sierra heartbeat format; ‘SpyHis’ which mimics the CTH Spyhis history output format; and ‘CSV’

Parameter	Value	Default
{=}	{= | are | is}	–
StreamTypes	{csv | original | spyhis}	–

Global

Syntax

Global [*Variables. . .*]

Summary

Define the global/reduction variables that should be written to the heartbeat database. The syntax is: "global {internal_name} as {DBname}".

Where {internal_name} is the name of the variable in the Sierra application; and {DBname} is the name as it should appear on the heartbeat database.

Labels

Syntax

Labels *{=} {off | on}*

Summary

Specifies whether labels will be displayed or just the value of the variable. Labels will be shown if this line is not present.

Parameter	Value	Default
{=}	{= | are | is}	–
Option	{off | on}	on

Legend

Syntax

Legend *{=} {off | on}*

Summary

Specifies whether a legend will be displayed prior to outputting any variables. The legend will not be shown unless this line is present. The legend shows the names of the variables that will be written to the heartbeat output stream. If the variable has multiple components, then the component count is shown after the variable e.g., velocity(3).

Parameter	Value	Default
{=}	{= | are | is}	–
Option	{off | on}	on

Monitor

Syntax

Monitor *{= | the} {history | restart | results}*

Summary

Specifies whether a line will be written to the heartbeat stream when either the results, history, and/or restart data are output.

Parameter	Value	Default
Equals	{= | the}	–
Option	{history | restart | results}	–

Nodal

Syntax

Nodal [*VariableList. . .*]

Summary

Define the nodal variables that should be written to the heartbeat database. The syntax is: "nodal {internal_name} at node {id} as {DBname}" or "nodal {internal_name} nearest location X, Y, Z as {DBname}".

Where {internal_name} is the name of the variable in the Sierra application; and {DBname} is the name as it should appear on the heartbeat database.

Node

Syntax

Node [*VariableList. . .*]

Summary

Define the nodal variables that should be written to the heartbeat database. The syntax is: "node {internal_name} at node {id} as {DBname}" or "node {internal_name} nearest location X, Y, Z as {DBname}".

Where {internal_name} is the name of the variable in the Sierra application; and {DBname} is the name as it should appear on the heartbeat database.

Nodeset

Syntax

Nodeset [*VariableList. . .*]

Summary

Define the nodeset variables that should be written to the heartbeat database. The syntax is: "nodeset {internal_name} at node {id} as {DBname}" or "nodeset {internal_name} nearest location X, Y, Z as {DBname}".

Where {internal_name} is the name of the variable in the Sierra application; and {DBname} is the name as it should appear on the heartbeat database. This

option finds a single value for the {internal_name} specified without having to specify a nodeset id or name.

Output On Signal

Syntax

Output On Signal *{=} {sigabrt | sigalrm | sigfpe | sighup | sigill | sigint | sigkill | sigpipe | sigquit | sigsegv | sigterm | sigusr1 | sigusr2}*

Summary

When the specified signal is raised, the output stream associated with this block will be output.

Parameter	Value	Default
{=}	{= | are | is}	–
Signals	{sigabrt | sigalrm | sigfpe | sighup | sigill | sigint | sigkill | sigpipe | sigquit | sigsegv | sigterm | sigusr1 | sigusr2}	–

Precision

Syntax

Precision *{=} Precision*

Summary

The precision to be used for the output of real variables (default=5).

Parameter	Value	Default
{=}	{= | are | is}	–
Precision	integer	5

Start Time

Syntax

Start Time *{=} Start_time*

Summary

Specify the time to start outputting results from this output request block. This time overrides all ‘at time’ and ‘at step’ specifications.

Parameter	Value	Default
{=}	{= | are | is}	–
Start_time	real	–

Stream Name

Syntax

Stream Name *{=}* *OutputFilename*

Summary

The filename of where the heartbeat data should be written. If the filename begins with the '/' character, it is an absolute path; otherwise, the path to the current directory will be prepended to the name. In addition, there are several predefined streams that can be specified. The predefined streams are 'cout' or 'stdout' specifies standard output; 'cerr', 'stderr', 'clog', or 'log' specifies standard error; 'output' or 'outputP0' specifies Sierra's standard output which is redirected to the file specified by the '-o' option on the command line. If the file already exists, it is overwritten. If this line is omitted, then a filename will be created from the basename of the input file with a ".hrt" suffix appended.

Parameter	Value	Default
{=}	{= | are | is}	–
OutputFilename	string	–

Synchronize Output

Syntax

Synchronize Output

Summary

In an analysis with multiple regions, it is sometimes desirable to synchronize the output of results data between the regions. This can be done by adding the SYNCHRONIZE OUTPUT command line to the results output block. If a results block has this set, then it will write output whenever a previous region writes output. The ordering of regions is based on the order in the input file, algorithmic considerations, or by solution control specifications.

Although the USE OUTPUT SCHEDULER command line can also synchronize output between regions, the SYNCHRONIZE OUTPUT command line will synchronize the output with regions where the output frequency is not under the direct control of the Sierra IO system. Examples of this are typically coupled applications where one or more of the codes are not Sierra-based applications such as Alegra and CTH. A results block with SYNCHRONIZE OUTPUT specified will also synchronize its output with the output of the external code.

The SYNCHRONIZE OUTPUT command can be used with other output scheduling commands such as time-based or step-based output specifications.

Termination Time

Syntax

Termination Time *{=}* *Final_time*

Summary

Specify the time to stop outputting results from this output request block.

Parameter	Value	Default
{=}	{= | are | is}	–
Final_time	real	–

Timestamp Format

Syntax

Timestamp Format

Summary

The format to be used for the timestamp. See 'man strftime' for more information.

Timestep Adjustment Interval

Syntax

Timestep Adjustment Interval *{=}* *Nsteps*

Summary

Specify the number of steps to 'look ahead' and adjust the timestep to ensure that the specified output times or simulation end time will be hit 'exactly'.

Parameter	Value	Default
{=}	{= | are | is}	–
Nsteps	integer	–

Use Output Scheduler

Syntax

Use Output Scheduler *Timer_name*

Summary

Associates a predefined output scheduler with this output block (results, restart, heartbeat, or history).

Parameter	Value	Default
Timer_name	string	–

Variable

Syntax

Variable *{=}* *{edge | element | face | global | nodal | node}* *Variable_list. . .*

Summary

Define the variables that should be written to the heartbeat output. The user can

request that the values of certain variables be output on the heartbeat line. These variables are limited to region and framework control data currently. The syntax is:

```
variable = {entity_type} {internal_name} at
           {entity_type} {entity_id}     as {external_name}
variable = {entity_type} {internal_name} nearest location
           {x,y,z} as {external_name}
```

For global variables, use:

```
variable = global {internal_name} [as {external_name}]
```

Where:

```
entity_type = node, element, face, edge, global
internal_name = Sierra variable name
entity_id = id of the node, element, face, edge that you want
        the specified variable output at.
external_name = name of variable on the database.
```

The names 'timestep', and 'time' can be specified as variables also. They are the current timestep and simulation time. This line can appear multiple times.

Parameter	Value	Default
{=}	{= | are | is}	–
Option	{edge | element | face | global | nodal | node}	–
Variable_list	string...	–

History Output

Scope

Aria Region, Input_Output Region

Summary

Describes the location and type of the output stream used for outputting history for the enclosing region.

```
begin History Output Label

   Additional Steps {=} List_of_steps...

   Additional Times {=} List_of_times...

   At Step n {increment | interval} {=} m
```

```
  At Time Dt1 {increment | interval} {=} Dt2

  Auto Output {all | element | global | nodal} User Defined Variables [␣
↪In UserOutputHistoryList...  ]

  Database Name {=} StreamName

  Database Type {=} {catalyst | catalyst_exodus | cgns | dof | dof_
↪exodus | exodus | exodusii | exonull | generated | genesis | null |␣
↪parallel_exodus | textmesh}

  Element [ VariableList...  ]

  Exists {=} {abort | add_suffix | append | overwrite}

  Face [ VariableList...  ]

  Flush Interval {=} Option

  Global [ Variables...  ]

  Nodal [ VariableList...  ]

  Node [ VariableList...  ]

  Nodeset [ VariableList...  ]

  Output On Signal {=} {sigabrt | sigalrm | sigfpe | sighup | sigill |␣
↪sigint | sigkill | sigpipe | sigquit | sigsegv | sigterm | sigusr1 |␣
↪sigusr2}

  Overwrite {=} {false | no | off | on | true | yes}

  Property PropertyName {=} PropertyValue

  Start Time {=} Start_time

  Synchronize Output

  Termination Time {=} Final_time

  Timestep Adjustment Interval {=} Nsteps

  Title
```

```
   Use Dynamic Topology Io

   Use Output Scheduler Timer_name

   Variable {=} {edge | element | face | global | nodal | node} Variable_
↪list...

end History Output Label
```

Line Commands

Additional Steps

Syntax

Additional Steps *{=} List_of_steps...*

Summary

Additional simulation steps when output should occur.

Parameter	Value	Default
{=}	{= | are | is}	–
List_of_steps	integer...	–

Additional Times

Syntax

Additional Times *{=} List_of_times...*

Summary

Additional simulation times when output should occur.

Parameter	Value	Default
{=}	{= | are | is}	–
List_of_times	real...	–

At Step

Syntax

At Step *n {increment | interval} {=} m*

Summary

Specify an output interval in terms of the internal iteration step count. The first step specifies the step count at the beginning of this interval and the second step specifies the output frequency to be used within this interval.

Parameter	Value	Default
n	integer	–
Option	{increment | interval}	–
{=}	{= | are | is}	–
m	integer	–

At Time

Syntax

At Time *Dt1 {increment | interval} {=} Dt2*

Summary

Specify an output interval in terms of the internal simulation time. The first time specifies the time at the beginning of this time interval and the second time specifies the output frequency to be used within this interval.

Parameter	Value	Default
Dt1	real	–
Option	{increment | interval}	–
{=}	{= | are | is}	–
Dt2	real	–

Auto Output

Syntax

Auto Output *{all | element | global | nodal}* User Defined Variables [In *UserOutputHistoryList. . .*]

Summary

Allows users to automatically output all user output defined variables for the type requested.

Parameter	Value	Default
auto_output_type_2	{all | element | global | nodal}	–

Database Name

Syntax

Database Name *{=} StreamName*

Summary

The base name of the database containing the output history. If the filename begins with the '/' character, it is an absolute path; otherwise, the path to the current directory will be prepended to the name. If this line is omitted, then a filename will be created from the basename of the input file with a ".h" suffix appended.

Parameter	Value	Default
{=}	{= | are | is}	–
StreamName	string	–

Database Type

Syntax

Database Type *{=} {catalyst | catalyst_exodus | cgns | dof | dof_exodus | exodus | exodusii | exonull | generated | genesis | null | parallel_exodus | textmesh}*

Summary

The database type/format to be used for the output history.

Parameter	Value	Default
{=}	{= | are | is}	–
DatabaseTypes	{catalyst | catalyst_exodus | cgns | dof | dof_exodus | exodus | exodusii | exonull | generated | genesis | null | parallel_exodus | textmesh}	–

Element

Syntax

Element [*VariableList. . .*]

Summary

Define the element variables that should be written to the history database. The syntax is: “element {internal_name} at element {id} as {DBname}” or “element {internal_name} nearest location X, Y, Z as {DBname}”.

Where {internal_name} is the name of the variable in the Sierra application; and {DBname} is the name as it should appear on the history database.

Exists

Syntax

Exists *{=} {abort | add_suffix | append | overwrite}*

Summary

Specify the behavior when creating this database and there is an existing file with the same name. The default behavior is “OVERWRITE” which deletes the existing file and creates a new file of the same name. “APPEND” will (if possible) append the new data to the end of the existing file. “ABORT” will print an error message and end the analysis. “ADD_SUFFIX” will add a -s???? suffix where the ???? is replaced by a sequential number starting at 0002.

Parameter	Value	Default
{=}	{= | is}	–
Option2	{abort | add_suffix | append | overwrite}	–

Face

Syntax

Face [*VariableList. . .*]

Summary

Define the face variables that should be written to the history database. The syntax is: "face {internal_name} at face {id} as {DBname}" or "face {internal_name} nearest location X, Y, Z as {DBname}".

Where {internal_name} is the name of the variable in the Sierra application; and {DBname} is the name as it should appear on the history database.

Flush Interval

Syntax

Flush Interval *{=}* *Option*

Summary

The minimum time interval (in seconds) at which output will be explicitly flushed to disk. The default is 10 seconds.

Parameter	Value	Default
{=}	{= | are | is}	–
Option	integer	10

Global

Syntax

Global [*Variables. . .*]

Summary

Define the global/reduction variables that should be written to the history database. The syntax is: "global {internal_name} as {DBname}".

Where {internal_name} is the name of the variable in the Sierra application; and {DBname} is the name as it should appear on the history database.

Nodal

Syntax

Nodal [*VariableList. . .*]

Summary

Define the nodal variables that should be written to the history database. The

syntax is: “nodal {internal_name} at node {id} as {DBname}” or “nodal {internal_name} nearest location X, Y, Z as {DBname}”.

Where {internal_name} is the name of the variable in the Sierra application; and {DBname} is the name as it should appear on the history database.

Node

Syntax

Node [*VariableList. . .*]

Summary

Define the nodal variables that should be written to the history database. The syntax is: “node {internal_name} at node {id} as {DBname}” or “node {internal_name} nearest location X, Y, Z as {DBname}”.

Where {internal_name} is the name of the variable in the Sierra application; and {DBname} is the name as it should appear on the history database.

Nodeset

Syntax

Nodeset [*VariableList. . .*]

Summary

Define the nodeset variables that should be written to the history database. The syntax is: “nodeset {internal_name} at node {id} as {DBname}” or “nodeset {internal_name} nearest location X, Y, Z as {DBname}”.

Where {internal_name} is the name of the variable in the Sierra application; and {DBname} is the name as it should appear on the history database.

Output On Signal

Syntax

Output On Signal *{=} {sigabrt | sigalrm | sigfpe | sighup | sigill | sigint | sigkill | sigpipe | sigquit | sigsegv | sigterm | sigusr1 | sigusr2}*

Summary

When the specified signal is raised, the output stream associated with this block will be output.

Parameter	Value	Default
{=}	{= | are | is}	–
Signals	{sigabrt | sigalrm | sigfpe | sighup | sigill | sigint | sigkill | sigpipe | sigquit | sigsegv | sigterm | sigusr1 | sigusr2}	–

Overwrite

Syntax

Overwrite *{=} {false | no | off | on | true | yes}*

Summary

(DEPRECATED, Use EXISTS) Specify whether the database should be overwritten if it exists. The default behavior is to overwrite unless this command is specified in the output block and either off, false, or no is specified.

Parameter	Value	Default
{=}	{= | is}	–
Option2	{false | no | off | on | true | yes}	–

Property

Syntax

Property *PropertyName {=} PropertyValue*

Summary

Define a database property named “PropertyName” with the value “PropertyValue”. If PropertyValue consists of all digits, it will define an integer property. If PropertyValue is “true” or “yes” or “false” or “no”, it will define a logical property; otherwise it will define a string property. Supported properties are typically database dependent; Some history-related properties are:

- VARIABLE_NAME_CASE = upper|lower
- MAX_NAME_LENGTH = value (32)

Parameter	Value	Default
PropertyName	string	–
{=}	{= | are | is}	–
PropertyValue	string	–

Start Time

Syntax

Start Time *{=} Start_time*

Summary

Specify the time to start outputting results from this output request block. This time overrides all ‘at time’ and ‘at step’ specifications.

Parameter	Value	Default
{=}	{= | are | is}	–
Start_time	real	–

Synchronize Output

Syntax

Synchronize Output

Summary

In an analysis with multiple regions, it is sometimes desirable to synchronize the output of results data between the regions. This can be done by adding the SYNCHRONIZE OUTPUT command line to the results output block. If a results block has this set, then it will write output whenever a previous region writes output. The ordering of regions is based on the order in the input file, algorithmic considerations, or by solution control specifications.

Although the USE OUTPUT SCHEDULER command line can also synchronize output between regions, the SYNCHRONIZE OUTPUT command line will synchronize the output with regions where the output frequency is not under the direct control of the Sierra IO system. Examples of this are typically coupled applications where one or more of the codes are not Sierra-based applications such as Alegra and CTH. A results block with SYNCHRONIZE OUTPUT specified will also synchronize its output with the output of the external code.

The SYNCHRONIZE OUTPUT command can be used with other output scheduling commands such as time-based or step-based output specifications.

Termination Time

Syntax

Termination Time *{=}* *Final_time*

Summary

Specify the time to stop outputting results from this output request block.

Parameter	Value	Default
{=}	{= | are | is}	–
Final_time	real	–

Timestep Adjustment Interval

Syntax

Timestep Adjustment Interval *{=}* *Nsteps*

Summary

Specify the number of steps to ‘look ahead’ and adjust the timestep to ensure that the specified output times or simulation end time will be hit ‘exactly’.

Parameter	Value	Default
{=}	{= | are | is}	–
Nsteps	integer	–

Title

Syntax

Title

Summary

Specify the title to be used for this specific output block.

Use Dynamic Topology Io

Syntax

Use Dynamic Topology Io

Summary

Specify that the app use IO for dynamic topology modifications where the output files are stored in a single database. Legacy file format for dynamically changing topology results in the creation of multiple files for each output on a mesh modification. This option leverages the ability of netCDF to create mesh groups within a single database and concatenate all mesh files into one. The names of each mesh group are of the form IOSS_MESH_GROUP-??? where ??? is the 1-based output index 1, 2, . . . , 10, ..., 100, . . . Please note that netCDF has a current limit of 65,536 groups

Use Output Scheduler

Syntax

Use Output Scheduler *Timer_name*

Summary

Associates a predefined output scheduler with this output block (results, restart, heartbeat, or history).

Parameter	Value	Default
Timer_name	string	–

Variable

Syntax

Variable *{=} {edge | element | face | global | nodal | node} Variable_list. . .*

Summary

Define the variables that should be written to the history database. The syntax is: “variable = entity {internal_name} at entity {id} as {DBname}” or “variable = entity {internal_name} nearest location X, Y, Z as {DBname}” or “variable = entity {internal_name} at location X, Y, Z as {DBname}”.

Where {entity} is ‘node’, ‘element’, ‘face’, or ‘edge’; {internal_name} is the name of the variable in the Sierra application; and {DBname} is the name as it should appear on the history database.

Parameter	Value	Default
{=}	{= | are | is}	–
Option	{edge | element | face | global | nodal | node}	–
Variable_list	string. . .	–

Output Scheduler

Scope

Sierra

Summary

Defines an output scheduler at the domain level which can be used by one or more output blocks (restart, results, heartbeat, history> at region scope. If used by multiple output blocks, their output will be synchronized.

```
begin Output Scheduler Label

   Additional Steps {=} List_of_steps...

   Additional Times {=} List_of_times...

   At Step n {increment | interval} {=} m

   At Time Dt1 {increment | interval} {=} Dt2

   Output On Signal {=} {sigabrt | sigalrm | sigfpe | sighup | sigill |␣
↪sigint | sigkill | sigpipe | sigquit | sigsegv | sigterm | sigusr1 |␣
↪sigusr2}

   Start Time {=} Start_time

   Synchronize Output

   Termination Time {=} Final_time

   Timestep Adjustment Interval {=} Nsteps

   Use Output Scheduler Timer_name

end Output Scheduler Label
```

Line Commands

Additional Steps

Syntax

Additional Steps *{=} List_of_steps. . .*

Summary

Additional simulation steps when output should occur.

Parameter	Value	Default
{=}	{= | are | is}	–
List_of_steps	integer. . .	–

Additional Times

Syntax

Additional Times *{=} List_of_times. . .*

Summary

Additional simulation times when output should occur.

Parameter	Value	Default
{=}	{= | are | is}	–
List_of_times	real. . .	–

At Step

Syntax

At Step *n {increment | interval} {=} m*

Summary

Specify an output interval in terms of the internal iteration step count. The first step specifies the step count at the beginning of this interval and the second step specifies the output frequency to be used within this interval.

Parameter	Value	Default
n	integer	–
Option	{increment | interval}	–
{=}	{= | are | is}	–
m	integer	–

At Time

Syntax

At Time *Dt1 {increment | interval} {=} Dt2*

Summary

Specify an output interval in terms of the internal simulation time. The first time specifies the time at the beginning of this time interval and the second time specifies the output frequency to be used within this interval.

Parameter	Value	Default
Dt1	real	–
Option	{increment | interval}	–
{=}	{= | are | is}	–
Dt2	real	–

Output On Signal

Syntax

Output On Signal *{=} {sigabrt | sigalrm | sigfpe | sighup | sigill | sigint | sigkill | sigpipe | sigquit | sigsegv | sigterm | sigusr1 | sigusr2}*

Summary

When the specified signal is raised, the output stream associated with this block will be output.

Parameter	Value	Default
{=}	{= | are | is}	–
Signals	{sigabrt | sigalrm | sigfpe | sighup | sigill | sigint | sigkill | sigpipe | sigquit | sigsegv | sigterm | sigusr1 | sigusr2}	–

Start Time

Syntax

Start Time *{=} Start_time*

Summary

Specify the time to start outputting results from this output request block. This time overrides all ‘at time’ and ‘at step’ specifications.

Parameter	Value	Default
{=}	{= | are | is}	–
Start_time	real	–

Synchronize Output

Syntax

Synchronize Output

Summary

In an analysis with multiple regions, it is sometimes desirable to synchronize the output of results data between the regions. This can be done by adding the SYNCHRONIZE OUTPUT command line to the results output block. If a results block has this set, then it will write output whenever a previous region writes output. The ordering of regions is based on the order in the input file, algorithmic considerations, or by solution control specifications.

Although the USE OUTPUT SCHEDULER command line can also synchronize output between regions, the SYNCHRONIZE OUTPUT command line will synchronize the output with regions where the output frequency is not under the direct control of the Sierra IO system. Examples of this are typically coupled applications where one or more of the codes are not Sierra-based applications such as Alegra and CTH. A results block with SYNCHRONIZE OUTPUT specified will also synchronize its output with the output of the external code.

The SYNCHRONIZE OUTPUT command can be used with other output scheduling commands such as time-based or step-based output specifications.

Termination Time

Syntax

Termination Time *{=}* *Final_time*

Summary

Specify the time to stop outputting results from this output request block.

Parameter	Value	Default
{=}	{= | are | is}	–
Final_time	real	–

Timestep Adjustment Interval

Syntax

Timestep Adjustment Interval *{=}* *Nsteps*

Summary

Specify the number of steps to ‘look ahead’ and adjust the timestep to ensure that the specified output times or simulation end time will be hit ‘exactly’.

Parameter	Value	Default
{=}	{= | are | is}	–
Nsteps	integer	–

Use Output Scheduler

Syntax

Use Output Scheduler *Timer_name*

Summary

Associates a predefined output scheduler with this output block (results, restart, heartbeat, or history).

Parameter	Value	Default
Timer_name	string	–

7.37 Postprocessing

A list of all postprocessing operations is provided below

Area

Block Style

Scope

Aria Region, Equation System, Explicit Equation System, Root Finder Equation System

Summary

Specify block-style post processing command

```
begin Postprocess Area

   Location = value ([Touching TouchingMeshExtent]  | [Opposing␣
↪OpposingMeshExtent]) [Contacting contacting surface]

   Output Name = value

end Postprocess Area
```

Line Commands

Location

Syntax

Location = *value* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) [Contacting *contacting surface*]

Summary

Name of the mesh block or entity to post process.

Parameter	Value	Default
value	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
contacting surface	string	–

Output Name

Syntax

Output Name = *value*

Summary

Name of the output variable or field.

Parameter	Value	Default
value	string	–

Equation

Syntax

Postprocess Area *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) [Contacting *contacting surface*] As *alias name*

Summary

Postprocess AREA of an expression.

Parameter	Value	Default
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
contacting surface	string	–
alias name	string	–

Average

Block Style

Scope

Aria Region, Equation System, Explicit Equation System, Root Finder Equation System

Summary

Specify block-style post processing command

```
begin Postprocess Average

   Weight Function = string_function

   Location = value ([Touching TouchingMeshExtent]  | [Opposing␣
↪OpposingMeshExtent]) [Contacting contacting surface]

   Output Name = value

   Source Type = Expression Name = source name ([{of} SpeciesName]  | [
↪{in} MaterialPhaseName]  | [{ls} {a | b | c}]) [Model = model name]

   Source Type = Scalar_String_Function F = string_function

   Source Type = Function string_function

   Source Type = Field Name = source name

end Postprocess Average
```

Line Commands

Weight Function

Syntax

Weight Function = *string_function*

Summary

Weighting function to use for averaging

Parameter	Value	Default
string_function	"string"	–

Location

Syntax

Location = *value* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) [Contacting *contacting surface*]

Summary

Name of the mesh block or entity to post process.

Parameter	Value	Default
value	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
contacting surface	string	–

Output Name

Syntax

Output Name = *value*

Summary

Name of the output variable or field.

Parameter	Value	Default
value	string	–

Source Type

Syntax

Source Type = Expression Name = *source name* ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) [Model = *model name*]

Summary

Postprocess an expression source type, takes optional species/material phase/model name arguments

Parameter	Value	Default
source name	string	–
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
model name	string	–

Source Type

Syntax

Source Type = Scalar_String_Function F = *string_function*

Summary

Postprocess using a scalar string function

Parameter	Value	Default
string_function	“string”	–

Source Type

Syntax

Source Type = Function *string_function*

Summary

Postprocess a scalar string function using a less verbose command

Parameter	Value	Default
string_function	"string"	–

Source Type

Syntax

Source Type = Field Name = *source name*

Summary

Postprocess a mesh field

Parameter	Value	Default
source name	string	–

Expression

Syntax

Postprocess Average Of Expression *src name* ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) [Contacting *contacting surface*] [Model *Model Name*] [With Weighting *weight function*] As *alias name*

Summary

Postprocess AVERAGE of an expression.

Parameter	Value	Default
src name	string	–
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
contacting surface	string	–
Model Name	string	–
weight function	"string"	–
alias name	string	–

Field

Syntax

Postprocess Average Of Field *src name {@ | at | for | on | over} Mesh Extent Name* As *alias name*

Parameter	Value	Default
src name	string	–
Mesh Extent Name	string	–
alias name	string	–

Function

Syntax

Postprocess Average Of Function *string_function {@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) [Contacting *contacting surface*] [With Weighting *weight function*] As *alias name*

Summary

Postprocess a scalar string function using a less verbose command

Parameter	Value	Default
string_function	"string"	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
contacting surface	string	–
weight function	"string"	–
alias name	string	–

Scalar_String_Function

Syntax

Postprocess Average Of Scalar_String_Function F = *string_function {@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) [Contacting *contacting surface*] [With Weighting *weight function*] As *alias name*

Summary

Postprocess using a scalar_string_function

Parameter	Value	Default
string_function	"string"	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
contacting surface	string	–
weight function	"string"	–
alias name	string	–

Averaged_Flux

Block Style

Scope

Aria Region, Equation System, Explicit Equation System, Root Finder Equation System

Summary

Specify block-style post processing command

`begin Postprocess Averaged_Flux`

```
  Weight Function = string_function

  Equation Name = value ([{of} SpeciesName]  | [{in} MaterialPhaseName] ␣
↪| [{ls} {a | b | c}])

  Location = value ([Touching TouchingMeshExtent]  | [Opposing␣
↪OpposingMeshExtent]) [Contacting contacting surface]

  Output Name = value

  Source Type = Expression Name = source name ([{of} SpeciesName]  | [
↪{in} MaterialPhaseName]  | [{ls} {a | b | c}]) [Model = model name]

  Source Type = Vector_String_Function (F_X = string_function_x  | F_Y =␣
↪string_function_y  | [F_Z = string_function_z])

end Postprocess Averaged_Flux
```

Line Commands

Weight Function

Syntax

Weight Function = *string_function*

Summary

Weighting function to use for averaging

Parameter	Value	Default
string_function	"string"	–

Equation Name

Syntax

Equation Name = *value* ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*])

Summary

Equation name to use for residual-based flux calculation

Parameter	Value	Default
value	string	–
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–

Location

Syntax

Location = *value* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) [Contacting *contacting surface*]

Summary

Name of the mesh block or entity to post process.

Parameter	Value	Default
value	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
contacting surface	string	–

Output Name

Syntax

Output Name = *value*

Summary

Name of the output variable or field.

Parameter	Value	Default
value	string	–

Source Type

Syntax

Source Type = Expression Name = *source name* ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) [Model = *model name*]

Summary

Postprocess an expression source type, takes optional species/material phase/model name arguments

Parameter	Value	Default
source name	string	–
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
model name	string	–

Source Type

Syntax

Source Type = Vector_String_Function (F_X = *string_function_x* | F_Y = *string_function_y* | [F_Z = *string_function_z*])

Summary

Postprocess using a vector string function

Parameter	Value	Default
string_function_x	"string"	–
string_function_y	"string"	–
string_function_z	"string"	–

Equation

Syntax

Postprocess Averaged_Flux *{of equation} equation name* ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) On *Mesh Extent Name...* As *alias name* [Multiplier *{=} multiplier*]

Summary

Postprocess the flux of an equation using the residual

Parameter	Value	Default
{of equation}	{for | of | of equation}	–
equation name	string	–
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string. . .	–
alias name	string	–
{=}	{= | are | is}	–
multiplier	real	–

Expression

Syntax

Postprocess Averaged_Flux Of Expression *src name* ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) [Contacting *contacting surface*] [Model *Model Name*] [With Weighting *weight function*] As *alias name*

Summary

Postprocess AVERAGED_FLUX of an expression.

Parameter	Value	Default
src name	string	–
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
contacting surface	string	–
Model Name	string	–
weight function	“string”	–
alias name	string	–

Vector_String_Function

Syntax

Postprocess Averaged_Flux Of Vector_String_Function (F_X = *string_function_x* | F_Y = *string_function_y* | [F_Z = *string_function_z*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) [Contacting *contacting surface*] [With Weighting *weight function*] As *alias name*

Summary

Postprocess using a vector_string_function

Parameter	Value	Default
string_function_x	"string"	–
string_function_y	"string"	–
string_function_z	"string"	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
contacting surface	string	–
weight function	"string"	–
alias name	string	–

Center

Gravity

Syntax

Postprocess Center_Of_Gravity *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) [Contacting *contacting surface*] As *alias name*

Summary

Postprocess CENTER_OF_GRAVITY of an expression.

Parameter	Value	Default
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
contacting surface	string	–
alias name	string	–

Flux_Scalar_Field

Block Style

Scope

Aria Region, Equation System, Explicit Equation System, Root Finder Equation System

Summary

Specify block-style post processing command

```
begin Postprocess Flux_Scalar_Field

  Equation Name = value ([{of} SpeciesName]  | [{in} MaterialPhaseName] ␣
↪| [{ls} {a | b | c}])

  Location = value

  Output Name = value

end Postprocess Flux_Scalar_Field
```

Line Commands

Equation Name

Syntax

Equation Name = *value* ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*])

Summary

Equation name to use for residual-based flux calculation

Parameter	Value	Default
value	string	–
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–

Location

Syntax

Location = *value*

Summary

Name of the mesh block or entity to post process.

Parameter	Value	Default
value	string	–

Output Name

Syntax

Output Name = *value*

Summary

Name of the output variable or field.

Parameter	Value	Default
value	string	–

Equation

Syntax

Postprocess Flux_Scalar_Field *{of equation}* *equation name* ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) On *Mesh Extent Name. . .* As *alias name* [Multiplier *{=}* *multiplier*]

Summary

Postprocess the flux of an equation using the residual

Parameter	Value	Default
{of equation}	{for | of | of equation}	–
equation name	string	–
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string. . .	–
alias name	string	–
{=}	{= | are | is}	–
multiplier	real	–

Flux_Vector_Field

Block Style

Scope

Aria Region, Equation System, Explicit Equation System, Root Finder Equation System

Summary

Specify block-style post processing command

```
begin Postprocess Flux_Vector_Field

  Equation Name = value ([{of} SpeciesName]  | [{in} MaterialPhaseName] ␣
↪| [{ls} {a | b | c}])

  Location = value

  Output Name = value

end Postprocess Flux_Vector_Field
```

Line Commands

Equation Name

Syntax

Equation Name = *value* ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*])

Summary

Equation name to use for residual-based flux calculation

Parameter	Value	Default
value	string	–
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–

Location

Syntax

Location = *value*

Summary

Name of the mesh block or entity to post process.

Parameter	Value	Default
value	string	–

Output Name

Syntax

Output Name = *value*

Summary

Name of the output variable or field.

Parameter	Value	Default
value	string	–

Equation

Syntax

Postprocess Flux_Vector_Field *{of equation}* *equation name* ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) On *Mesh Extent Name. . .* As *alias name* [Multiplier *{=}* *multiplier*]

Summary

Postprocess the flux of an equation using the residual

Parameter	Value	Default
{of equation}	{for | of | of equation}	–
equation name	string	–
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string. . .	–
alias name	string	–
{=}	{= | are | is}	–
multiplier	real	–

Global

Block Style

Scope

Aria Region, Equation System, Explicit Equation System, Root Finder Equation System

Summary

Specify block-style post-processing command

```
begin Postprocess Global

   Function = string_function

   Output Name = value

end Postprocess Global
```

Line Commands

Function

Syntax

Function = *string_function*

Summary

Function to postprocess.

Parameter	Value	Default
string_function	"string"	–

Output Name

Syntax

Output Name = *value*

Summary

Name of the output global variable.

Parameter	Value	Default
value	string	–

Line Style

Syntax

Postprocess Global_Function *string_function* As *alias name*

Summary

Postprocess global variables using a string function

Parameter	Value	Default
string_function	"string"	–
alias name	string	–

H_1_Norm

Block Style

Scope

Aria Region, Equation System, Explicit Equation System, Root Finder Equation System

Summary

Specify block-style post processing command

```
begin Postprocess H_1_Norm

  Location = value ([Touching TouchingMeshExtent]  | [Opposing␣
↪OpposingMeshExtent]) [Contacting contacting surface]

  Output Name = value

  Source Type = Expression Name = source name ([{of} SpeciesName]  | [
↪{in} MaterialPhaseName]  | [{ls} {a | b | c}]) [Model = model name]

  Source Type = Scalar_String_Function F = string_function

  Source Type = Function string_function

end Postprocess H_1_Norm
```

Line Commands

Location

Syntax

Location = *value* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) [Contacting *contacting surface*]

Summary

Name of the mesh block or entity to post process.

Parameter	Value	Default
value	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
contacting surface	string	–

Output Name

Syntax

Output Name = *value*

Summary

Name of the output variable or field.

Parameter	Value	Default
value	string	–

Source Type

Syntax

Source Type = Expression Name = *source name* ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) [Model = *model name*]

Summary

Postprocess an expression source type, takes optional species/material phase/model name arguments

Parameter	Value	Default
source name	string	–
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
model name	string	–

Source Type

Syntax

Source Type = Scalar_String_Function F = *string_function*

Summary

Postprocess using a scalar string function

Parameter	Value	Default
string_function	"string"	–

Source Type

Syntax

Source Type = Function *string_function*

Summary

Postprocess a scalar string function using a less verbose command

Parameter	Value	Default
string_function	"string"	–

Expression

Syntax

Postprocess H_1_Norm Of Expression *src name* ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) [Contacting *contacting surface*] [Model *Model Name*] As *alias name*

Summary

Postprocess H_1_NORM of an expression.

Parameter	Value	Default
src name	string	–
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
contacting surface	string	–
Model Name	string	–
alias name	string	–

Function

Syntax

Postprocess H_1_Norm Of Function *string_function {@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) [Contacting *contacting surface*] [With Weighting *weight function*] As *alias name*

Summary

Postprocess a scalar string function using a less verbose command

Parameter	Value	Default
string_function	"string"	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
contacting surface	string	–
weight function	"string"	–
alias name	string	–

Scalar_String_Function

Syntax

Postprocess H_1_Norm Of Scalar_String_Function F = *string_function {@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) [Contacting *contacting surface*] [With Weighting *weight function*] As *alias name*

Summary

Postprocess using a scalar_string_function

Parameter	Value	Default
string_function	"string"	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
contacting surface	string	–
weight function	"string"	–
alias name	string	–

Integral

Block Style

Scope

Aria Region, Equation System, Explicit Equation System, Root Finder Equation System

Summary

Specify block-style post processing command

```
begin Postprocess Integral

  Location = value ([Touching TouchingMeshExtent]  | [Opposing␣
↪OpposingMeshExtent]) [Contacting contacting surface]

  Output Name = value

  Source Type = Expression Name = source name ([{of} SpeciesName]  | [
↪{in} MaterialPhaseName]  | [{ls} {a | b | c}]) [Model = model name]

  Source Type = Scalar_String_Function F = string_function

  Source Type = Function string_function
```

```
end Postprocess Integral
```

Line Commands

Location

Syntax

Location = *value* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) [Contacting *contacting surface*]

Summary

Name of the mesh block or entity to post process.

Parameter	Value	Default
value	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
contacting surface	string	–

Output Name

Syntax

Output Name = *value*

Summary

Name of the output variable or field.

Parameter	Value	Default
value	string	–

Source Type

Syntax

Source Type = Expression Name = *source name* ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) [Model = *model name*]

Summary

Postprocess an expression source type, takes optional species/material phase/model name arguments

Parameter	Value	Default
source name	string	–
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
model name	string	–

Source Type

Syntax

Source Type = Scalar_String_Function F = *string_function*

Summary

Postprocess using a scalar string function

Parameter	Value	Default
string_function	"string"	–

Source Type

Syntax

Source Type = Function *string_function*

Summary

Postprocess a scalar string function using a less verbose command

Parameter	Value	Default
string_function	"string"	–

Expression

Syntax

Postprocess Integral Of Expression *src name* ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) [Contacting *contacting surface*] [Model *Model Name*] As *alias name*

Summary

Postprocess INTEGRAL of an expression.

Parameter	Value	Default
src name	string	–
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
contacting surface	string	–
Model Name	string	–
alias name	string	–

Function

Syntax

Postprocess Integral Of Function *string_function {@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) [Contacting *contacting surface*] [With Weighting *weight function*] As *alias name*

Summary

Postprocess a scalar string function using a less verbose command

Parameter	Value	Default
string_function	"string"	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
contacting surface	string	–
weight function	"string"	–
alias name	string	–

Scalar_String_Function

Syntax

Postprocess Integral Of Scalar_String_Function F = *string_function {@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) [Contacting *contacting surface*] [With Weighting *weight function*] As *alias name*

Summary

Postprocess using a scalar_string_function

Parameter	Value	Default
string_function	"string"	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
contacting surface	string	–
weight function	"string"	–
alias name	string	–

Integrated_Flux

Block Style

Scope

Aria Region, Equation System, Explicit Equation System, Root Finder Equation System

Summary

Specify block-style post processing command

```
begin Postprocess Integrated_Flux

  Equation Name = value ([{of} SpeciesName]  | [{in} MaterialPhaseName] ␣
↪| [{ls} {a | b | c}])

  Location = value ([Touching TouchingMeshExtent]  | [Opposing␣
↪OpposingMeshExtent]) [Contacting contacting surface]

  Output Name = value

  Source Type = Expression Name = source name ([{of} SpeciesName]  | [
↪{in} MaterialPhaseName]  | [{ls} {a | b | c}]) [Model = model name]
```

```
   Source Type = Vector_String_Function (F_X = string_function_x  | F_Y =␣
↪string_function_y  | [F_Z = string_function_z])

end Postprocess Integrated_Flux
```

Line Commands

Equation Name

Syntax

Equation Name = *value* ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*])

Summary

Equation name to use for residual-based flux calculation

Parameter	Value	Default
value	string	–
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–

Location

Syntax

Location = *value* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) [Contacting *contacting surface*]

Summary

Name of the mesh block or entity to post process.

Parameter	Value	Default
value	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
contacting surface	string	–

Output Name

Syntax

Output Name = *value*

Summary

Name of the output variable or field.

Parameter	Value	Default
value	string	–

Source Type

Syntax

Source Type = Expression Name = *source name* ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) [Model = *model name*]

Summary

Postprocess an expression source type, takes optional species/material phase/model name arguments

Parameter	Value	Default
source name	string	–
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
model name	string	–

Source Type

Syntax

Source Type = Vector_String_Function (F_X = *string_function_x* | F_Y = *string_function_y* | [F_Z = *string_function_z*])

Summary

Postprocess using a vector string function

Parameter	Value	Default
string_function_x	“string”	–
string_function_y	“string”	–
string_function_z	“string”	–

Equation

Syntax

Postprocess Integrated_Flux *{of equation}* *equation name* ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) On *Mesh Extent Name...* As *alias name* [Multiplier *{=}* *multiplier*]

Summary

Postprocess the flux of an equation using the residual

Parameter	Value	Default
{of equation}	{for | of | of equation}	–
equation name	string	–
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string...	–
alias name	string	–
{=}	{= | are | is}	–
multiplier	real	–

Expression

Syntax

Postprocess Integrated_Flux Of Expression *src name* ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* *Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) [Contacting *contacting surface*] [Model *Model Name*] As *alias name*

Summary

Postprocess INTEGRATED_FLUX of an expression.

Parameter	Value	Default
src name	string	–
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
contacting surface	string	–
Model Name	string	–
alias name	string	–

Vector_String_Function

Syntax

Postprocess Integrated_Flux Of Vector_String_Function (F_X = *string_function_x* | F_Y = *string_function_y* | [F_Z = *string_function_z*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) [Contacting *contacting surface*] [With Weighting *weight function*] As *alias name*

Summary

Postprocess using a vector_string_function

Parameter	Value	Default
string_function_x	"string"	–
string_function_y	"string"	–
string_function_z	"string"	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
contacting surface	string	–
weight function	"string"	–
alias name	string	–

L_1_Norm

Block Style

Scope

Aria Region, Equation System, Explicit Equation System, Root Finder Equation System

Summary

Specify block-style post processing command

```
begin Postprocess L_1_Norm

  Weight Function = string_function

  Location = value ([Touching TouchingMeshExtent]  | [Opposing␣
↪OpposingMeshExtent]) [Contacting contacting surface]

  Output Name = value

  Source Type = Expression Name = source name ([{of} SpeciesName]  | [
↪{in} MaterialPhaseName]  | [{ls} {a | b | c}]) [Model = model name]

  Source Type = Scalar_String_Function F = string_function

  Source Type = Function string_function

  Source Type = Field Name = source name

end Postprocess L_1_Norm
```

Line Commands

Weight Function

Syntax

Weight Function = *string_function*

Summary

Weighting function to use for averaging

Parameter	Value	Default
string_function	"string"	–

Location

Syntax

Location = *value* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) [Contacting *contacting surface*]

Summary

Name of the mesh block or entity to post process.

Parameter	Value	Default
value	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
contacting surface	string	–

Output Name

Syntax

Output Name = *value*

Summary

Name of the output variable or field.

Parameter	Value	Default
value	string	–

Source Type

Syntax

Source Type = Expression Name = *source name* ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) [Model = *model name*]

Summary

Postprocess an expression source type, takes optional species/material phase/model name arguments

Parameter	Value	Default
source name	string	–
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
model name	string	–

Source Type

Syntax

Source Type = Scalar_String_Function F = *string_function*

Summary

Postprocess using a scalar string function

Parameter	Value	Default
string_function	"string"	–

Source Type

Syntax

Source Type = Function *string_function*

Summary

Postprocess a scalar string function using a less verbose command

Parameter	Value	Default
string_function	"string"	–

Source Type

Syntax

Source Type = Field Name = *source name*

Summary

Postprocess a mesh field

Parameter	Value	Default
source name	string	–

Expression

Syntax

Postprocess L_1_Norm Of Expression *src name* ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) [Contacting *contacting surface*] [Model *Model Name*] [With Weighting *weight function*] As *alias name*

Summary

Postprocess L_1_NORM of an expression.

Parameter	Value	Default
src name	string	–
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
contacting surface	string	–
Model Name	string	–
weight function	"string"	–
alias name	string	–

Field

Syntax

Postprocess L_1_Norm Of Field *src name {@ | at | for | on | over} Mesh Extent Name* As *alias name*

Parameter	Value	Default
src name	string	–
Mesh Extent Name	string	–
alias name	string	–

Function

Syntax

Postprocess L_1_Norm Of Function *string_function {@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) [Contacting *contacting surface*] [With Weighting *weight function*] As *alias name*

Summary

Postprocess a scalar string function using a less verbose command

Parameter	Value	Default
string_function	"string"	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
contacting surface	string	–
weight function	"string"	–
alias name	string	–

Scalar_String_Function

Syntax

Postprocess L_1_Norm Of Scalar_String_Function F = *string_function {@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) [Contacting *contacting surface*] [With Weighting *weight function*] As *alias name*

Summary

Postprocess using a scalar_string_function

Parameter	Value	Default
string_function	"string"	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
contacting surface	string	–
weight function	"string"	–
alias name	string	–

L_2_Norm

Block Style

Scope

Aria Region, Equation System, Explicit Equation System, Root Finder Equation System

Summary

Specify block-style post processing command

`begin Postprocess L_2_Norm`

```
  Location = value ([Touching TouchingMeshExtent]  | [Opposing␣
↪OpposingMeshExtent]) [Contacting contacting surface]

  Output Name = value

  Source Type = Expression Name = source name ([{of} SpeciesName]  | [
↪{in} MaterialPhaseName]  | [{ls} {a | b | c}]) [Model = model name]

  Source Type = Scalar_String_Function F = string_function

  Source Type = Function string_function

  Source Type = Field Name = source name

end Postprocess L_2_Norm
```

Line Commands

Location

Syntax

Location = *value* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) [Contacting *contacting surface*]

Summary

Name of the mesh block or entity to post process.

Parameter	Value	Default
value	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
contacting surface	string	–

Output Name

Syntax

Output Name = *value*

Summary

Name of the output variable or field.

Parameter	Value	Default
value	string	–

Source Type

Syntax

Source Type = Expression Name = *source name* ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) [Model = *model name*]

Summary

Postprocess an expression source type, takes optional species/material phase/model name arguments

Parameter	Value	Default
source name	string	–
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
model name	string	–

Source Type

Syntax

Source Type = Scalar_String_Function F = *string_function*

Summary

Postprocess using a scalar string function

Parameter	Value	Default
string_function	"string"	–

Source Type

Syntax

Source Type = Function *string_function*

Summary

Postprocess a scalar string function using a less verbose command

Parameter	Value	Default
string_function	"string"	–

Source Type

Syntax

Source Type = Field Name = *source name*

Summary

Postprocess a mesh field

Parameter	Value	Default
source name	string	–

Expression

Syntax

Postprocess L_2_Norm Of Expression *src name* ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) [Contacting *contacting surface*] [Model *Model Name*] As *alias name*

Summary

Postprocess L_2_NORM of an expression.

Parameter	Value	Default
src name	string	–
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
contacting surface	string	–
Model Name	string	–
alias name	string	–

Field

Syntax

Postprocess L_2_Norm Of Field *src name {@ | at | for | on | over} Mesh Extent Name* As *alias name*

Parameter	Value	Default
src name	string	–
Mesh Extent Name	string	–
alias name	string	–

Function

Syntax

Postprocess L_2_Norm Of Function *string_function {@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) [Contacting *contacting surface*] [With Weighting *weight function*] As *alias name*

Summary

Postprocess a scalar string function using a less verbose command

Parameter	Value	Default
string_function	"string"	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
contacting surface	string	–
weight function	"string"	–
alias name	string	–

Scalar_String_Function

Syntax

Postprocess L_2_Norm Of Scalar_String_Function F = *string_function {@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) [Contacting *contacting surface*] [With Weighting *weight function*] As *alias name*

Summary

Postprocess using a scalar_string_function

Parameter	Value	Default
string_function	"string"	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
contacting surface	string	–
weight function	"string"	–
alias name	string	–

L_Inf_Norm

Block Style

Scope

Aria Region, Equation System, Explicit Equation System, Root Finder Equation System

Summary

Specify block-style post processing command

```
begin Postprocess L_Inf_Norm

  Location = value ([Touching TouchingMeshExtent]  | [Opposing␣
↪OpposingMeshExtent]) [Contacting contacting surface]

  Output Name = value

  Source Type = Expression Name = source name ([{of} SpeciesName]  | [
↪{in} MaterialPhaseName]  | [{ls} {a | b | c}]) [Model = model name]

  Source Type = Scalar_String_Function F = string_function

  Source Type = Function string_function

  Source Type = Field Name = source name

end Postprocess L_Inf_Norm
```

Line Commands

Location

Syntax

Location = *value* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) [Contacting *contacting surface*]

Summary

Name of the mesh block or entity to post process.

Parameter	Value	Default
value	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
contacting surface	string	–

Output Name

Syntax

Output Name = *value*

Summary

Name of the output variable or field.

Parameter	Value	Default
value	string	–

Source Type

Syntax

Source Type = Expression Name = *source name* ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) [Model = *model name*]

Summary

Postprocess an expression source type, takes optional species/material phase/model name arguments

Parameter	Value	Default
source name	string	–
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
model name	string	–

Source Type

Syntax

Source Type = Scalar_String_Function F = *string_function*

Summary

Postprocess using a scalar string function

Parameter	Value	Default
string_function	“string”	–

Source Type

Syntax

Source Type = Function *string_function*

Summary

Postprocess a scalar string function using a less verbose command

Parameter	Value	Default
string_function	"string"	–

Source Type

Syntax

Source Type = Field Name = *source name*

Summary

Postprocess a mesh field

Parameter	Value	Default
source name	string	–

Expression

Syntax

Postprocess L_Inf_Norm Of Expression *src name* ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) [Contacting *contacting surface*] [Model *Model Name*] As *alias name*

Summary

Postprocess L_INF_NORM of an expression.

Parameter	Value	Default
src name	string	–
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
contacting surface	string	–
Model Name	string	–
alias name	string	–

Field

Syntax

Postprocess L_Inf_Norm Of Field *src name {@ | at | for | on | over} Mesh Extent Name* As *alias name*

Parameter	Value	Default
src name	string	–
Mesh Extent Name	string	–
alias name	string	–

Function

Syntax

Postprocess L_Inf_Norm Of Function *string_function {@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) [Contacting *contacting surface*] [With Weighting *weight function*] As *alias name*

Summary

Postprocess a scalar string function using a less verbose command

Parameter	Value	Default
string_function	"string"	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
contacting surface	string	–
weight function	"string"	–
alias name	string	–

Scalar_String_Function

Syntax

Postprocess L_Inf_Norm Of Scalar_String_Function F = *string_function {@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) [Contacting *contacting surface*] [With Weighting *weight function*] As *alias name*

Summary

Postprocess using a scalar_string_function

Parameter	Value	Default
string_function	"string"	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
contacting surface	string	–
weight function	"string"	–
alias name	string	–

Mass

Block Style

Scope

Aria Region, Equation System, Explicit Equation System, Root Finder Equation System

Summary

Specify block-style post processing command

```
begin Postprocess Mass

  Location = value ([Touching TouchingMeshExtent]  | [Opposing␣
↪OpposingMeshExtent]) [Contacting contacting surface]

  Output Name = value

end Postprocess Mass
```

Line Commands

Location

Syntax

Location = *value* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) [Contacting *contacting surface*]

Summary

Name of the mesh block or entity to post process.

Parameter	Value	Default
value	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
contacting surface	string	–

Output Name

Syntax

Output Name = *value*

Summary

Name of the output variable or field.

Parameter	Value	Default
value	string	–

Equation

Syntax

Postprocess Mass *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) [Contacting *contacting surface*] As *alias name*

Summary

Postprocess MASS of an expression.

Parameter	Value	Default
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
contacting surface	string	–
alias name	string	–

Max

Block Style

Scope

Aria Region, Equation System, Explicit Equation System, Root Finder Equation System

Summary

Specify block-style post processing command

```
begin Postprocess Max

  Location = value ([Touching TouchingMeshExtent]  | [Opposing␣
↪OpposingMeshExtent]) [Contacting contacting surface]

  Output Name = value

  Source Type = Expression Name = source name ([{of} SpeciesName]  | [
↪{in} MaterialPhaseName]  | [{ls} {a | b | c}]) [Model = model name]

  Source Type = Scalar_String_Function F = string_function

  Source Type = Function string_function

  Source Type = Field Name = source name

end Postprocess Max
```

Line Commands

Location

Syntax

Location = *value* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) [Contacting *contacting surface*]

Summary

Name of the mesh block or entity to post process.

Parameter	Value	Default
value	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
contacting surface	string	–

Output Name

Syntax

Output Name = *value*

Summary

Name of the output variable or field.

Parameter	Value	Default
value	string	–

Source Type

Syntax

Source Type = Expression Name = *source name* ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) [Model = *model name*]

Summary

Postprocess an expression source type, takes optional species/material phase/model name arguments

Parameter	Value	Default
source name	string	–
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
model name	string	–

Source Type

Syntax

Source Type = Scalar_String_Function F = *string_function*

Summary

Postprocess using a scalar string function

Parameter	Value	Default
string_function	“string”	–

Source Type

Syntax

Source Type = Function *string_function*

Summary

Postprocess a scalar string function using a less verbose command

Parameter	Value	Default
string_function	“string”	–

Source Type

Syntax

Source Type = Field Name = *source name*

Summary

Postprocess a mesh field

Parameter	Value	Default
source name	string	–

Expression

Syntax

Postprocess Max Of Expression *src name* ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) [Contacting *contacting surface*] [Model *Model Name*] As *alias name*

Summary

Postprocess MAX of an expression.

Parameter	Value	Default
src name	string	–
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
contacting surface	string	–
Model Name	string	–
alias name	string	–

Field

Syntax

Postprocess Max Of Field *src name {@ | at | for | on | over} Mesh Extent Name* As *alias name*

Parameter	Value	Default
src name	string	–
Mesh Extent Name	string	–
alias name	string	–

Function

Syntax

Postprocess Max Of Function *string_function {@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) [Contacting *contacting surface*] [With Weighting *weight function*] As *alias name*

Summary

Postprocess a scalar string function using a less verbose command

Parameter	Value	Default
string_function	"string"	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
contacting surface	string	–
weight function	"string"	–
alias name	string	–

Scalar_String_Function

Syntax

Postprocess Max Of Scalar_String_Function F = *string_function {@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) [Contacting *contacting surface*] [With Weighting *weight function*] As *alias name*

Summary

Postprocess using a scalar_string_function

Parameter	Value	Default
string_function	"string"	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
contacting surface	string	–
weight function	"string"	–
alias name	string	–

Min

Block Style

Scope

Aria Region, Equation System, Explicit Equation System, Root Finder Equation System

Summary

Specify block-style post processing command

```
begin Postprocess Min

  Location = value ([Touching TouchingMeshExtent]  | [Opposing␣
↪OpposingMeshExtent]) [Contacting contacting surface]

  Output Name = value

  Source Type = Expression Name = source name ([{of} SpeciesName]  | [
↪{in} MaterialPhaseName]  | [{ls} {a | b | c}]) [Model = model name]

  Source Type = Scalar_String_Function F = string_function

  Source Type = Function string_function

  Source Type = Field Name = source name

end Postprocess Min
```

Line Commands

Location

Syntax

Location = *value* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) [Contacting *contacting surface*]

Summary

Name of the mesh block or entity to post process.

Parameter	Value	Default
value	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
contacting surface	string	–

Output Name

Syntax

Output Name = *value*

Summary

Name of the output variable or field.

Parameter	Value	Default
value	string	–

Source Type

Syntax

Source Type = Expression Name = *source name* ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) [Model = *model name*]

Summary

Postprocess an expression source type, takes optional species/material phase/model name arguments

Parameter	Value	Default
source name	string	–
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
model name	string	–

Source Type

Syntax

Source Type = Scalar_String_Function F = *string_function*

Summary

Postprocess using a scalar string function

Parameter	Value	Default
string_function	"string"	–

Source Type

Syntax

Source Type = Function *string_function*

Summary

Postprocess a scalar string function using a less verbose command

Parameter	Value	Default
string_function	"string"	–

Source Type

Syntax

Source Type = Field Name = *source name*

Summary

Postprocess a mesh field

Parameter	Value	Default
source name	string	–

Expression

Syntax

Postprocess Min Of Expression *src name* ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) [Contacting *contacting surface*] [Model *Model Name*] As *alias name*

Summary

Postprocess MIN of an expression.

Parameter	Value	Default
src name	string	–
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
contacting surface	string	–
Model Name	string	–
alias name	string	–

Field

Syntax

Postprocess Min Of Field *src name {@ | at | for | on | over} Mesh Extent Name* As *alias name*

Parameter	Value	Default
src name	string	–
Mesh Extent Name	string	–
alias name	string	–

Function

Syntax

Postprocess Min Of Function *string_function {@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) [Contacting *contacting surface*] [With Weighting *weight function*] As *alias name*

Summary

Postprocess a scalar string function using a less verbose command

Parameter	Value	Default
string_function	"string"	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
contacting surface	string	–
weight function	"string"	–
alias name	string	–

Scalar_String_Function

Syntax

Postprocess Min Of Scalar_String_Function F = *string_function {@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) [Contacting *contacting surface*] [With Weighting *weight function*] As *alias name*

Summary

Postprocess using a scalar_string_function

Parameter	Value	Default
string_function	"string"	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
contacting surface	string	–
weight function	"string"	–
alias name	string	–

Moment

Inertia

Syntax

Postprocess Moment_Of_Inertia *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) [Contacting *contacting surface*] As *alias name*

Summary

Postprocess MOMENT_OF_INERTIA of an expression.

Parameter	Value	Default
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
contacting surface	string	–
alias name	string	–

Rms

Block Style

Scope

Aria Region, Equation System, Explicit Equation System, Root Finder Equation System

Summary

Specify block-style post processing command

```
begin Postprocess Rms

  Weight Function = string_function

  Location = value ([Touching TouchingMeshExtent]  | [Opposing␣
↪OpposingMeshExtent]) [Contacting contacting surface]

  Output Name = value

  Source Type = Expression Name = source name ([{of} SpeciesName]  | [
↪{in} MaterialPhaseName]  | [{ls} {a | b | c}]) [Model = model name]

  Source Type = Scalar_String_Function F = string_function

  Source Type = Function string_function

  Source Type = Field Name = source name

end Postprocess Rms
```

Line Commands

Weight Function

Syntax

Weight Function = *string_function*

Summary

Weighting function to use for averaging

Parameter	Value	Default
string_function	"string"	–

Location

Syntax

Location = *value* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) [Contacting *contacting surface*]

Summary

Name of the mesh block or entity to post process.

Parameter	Value	Default
value	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
contacting surface	string	–

Output Name

Syntax

Output Name = *value*

Summary

Name of the output variable or field.

Parameter	Value	Default
value	string	–

Source Type

Syntax

Source Type = Expression Name = *source name* ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) [Model = *model name*]

Summary

Postprocess an expression source type, takes optional species/material phase/model name arguments

Parameter	Value	Default
source name	string	–
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
model name	string	–

Source Type

Syntax

Source Type = Scalar_String_Function F = *string_function*

Summary

Postprocess using a scalar string function

Parameter	Value	Default
string_function	"string"	–

Source Type

Syntax

Source Type = Function *string_function*

Summary

Postprocess a scalar string function using a less verbose command

Parameter	Value	Default
string_function	"string"	–

Source Type

Syntax

Source Type = Field Name = *source name*

Summary

Postprocess a mesh field

Parameter	Value	Default
source name	string	–

Expression

Syntax

Postprocess Rms Of Expression *src name* ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) [Contacting *contacting surface*] [Model *Model Name*] [With Weighting *weight function*] As *alias name*

Summary

Postprocess RMS of an expression.

Parameter	Value	Default
src name	string	–
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
contacting surface	string	–
Model Name	string	–
weight function	“string”	–
alias name	string	–

Field

Syntax

Postprocess Rms Of Field *src name {@ | at | for | on | over} Mesh Extent Name* As *alias name*

Parameter	Value	Default
src name	string	–
Mesh Extent Name	string	–
alias name	string	–

Function

Syntax

Postprocess Rms Of Function *string_function {@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) [Contacting *contacting surface*] [With Weighting *weight function*] As *alias name*

Summary

Postprocess a scalar string function using a less verbose command

Parameter	Value	Default
string_function	"string"	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
contacting surface	string	–
weight function	"string"	–
alias name	string	–

Scalar_String_Function

Syntax

Postprocess Rms Of Scalar_String_Function F = *string_function {@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) [Contacting *contacting surface*] [With Weighting *weight function*] As *alias name*

Summary

Postprocess using a scalar_string_function

Parameter	Value	Default
string_function	"string"	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
contacting surface	string	–
weight function	"string"	–
alias name	string	–

Value

Block Style

Scope

Aria Region, Equation System, Explicit Equation System, Root Finder Equation System

Summary

Specify block-style post processing command

begin Postprocess Value

```
  Weight Function = string_function

  Location = value ([Touching TouchingMeshExtent]  | [Opposing␣
↪OpposingMeshExtent]) [Contacting contacting surface]

  Output Name = value

  Source Type = Expression Name = source name ([{of} SpeciesName]  | [
↪{in} MaterialPhaseName]  | [{ls} {a | b | c}]) [Model = model name]␣
↪[Using element type] [Averaging Method {=} averaging method]

  Source Type = Scalar_String_Function F = string_function [Using␣
↪element type] [Averaging Method {=} averaging method]

  Source Type = Function string_function [Using element type] [Averaging␣
↪Method {=} averaging method]

  Source Type = Vector_String_Function (F_X = string_function_x  | F_Y␣
↪= string_function_y  | [F_Z = string_function_z]) [Using element type]␣
↪[Averaging Method {=} averaging method]

  Source Type = Tensor_String_Function (F_Xx = string_function_xx  | F_
↪Xy = string_function_xy  | F_Yx = string_function_yx  | F_Yy = string_
↪function_yy  | [F_Zx = string_function_zx]  | [F_Zy = string_function_
↪zy]  | [F_Xz = string_function_xz]  | [F_Yz = string_function_yz]  |␣
↪[F_Zz = string_function_zz]) [Using element type] [Averaging Method {=}␣
↪averaging method]
```

end Postprocess Value

Line Commands

Weight Function

Syntax

Weight Function = *string_function*

Summary

Weighting function to use for averaging

Parameter	Value	Default
string_function	"string"	–

Location

Syntax

Location = *value* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) [Contacting *contacting surface*]

Summary

Name of the mesh block or entity to post process.

Parameter	Value	Default
value	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
contacting surface	string	–

Output Name

Syntax

Output Name = *value*

Summary

Name of the output variable or field.

Parameter	Value	Default
value	string	–

Source Type

Syntax

Source Type = Expression Name = *source name* ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) [Model = *model name*] [Using *element type*] [Averaging Method *{=}* *averaging method*]

Summary

Postprocess an expression source type, takes optional species/material phase/model name arguments

Parameter	Value	Default
source name	string	–
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
model name	string	–
element type	string	–
{=}	{= | are | is}	–
averaging method	string	–

Source Type

Syntax

Source Type = Scalar_String_Function F = *string_function* [Using *element type*] [Averaging Method *{=} averaging method*]

Summary

Postprocess using a scalar string function

Parameter	Value	Default
string_function	"string"	–
element type	string	–
{=}	{= | are | is}	–
averaging method	string	–

Source Type

Syntax

Source Type = Function *string_function* [Using *element type*] [Averaging Method *{=} averaging method*]

Summary

Postprocess a scalar string function using a less verbose command

Parameter	Value	Default
string_function	"string"	–
element type	string	–
{=}	{= | are | is}	–
averaging method	string	–

Source Type

Syntax

Source Type = Vector_String_Function (F_X = *string_function_x* | F_Y = *string_function_y* | [F_Z = *string_function_z*]) [Using *element type*] [Averaging Method *{=}* *averaging method*]

Summary

Postprocess using a vector string function

Parameter	Value	Default
string_function_x	"string"	–
string_function_y	"string"	–
string_function_z	"string"	–
element type	string	–
{=}	{= | are | is}	–
averaging method	string	–

Source Type

Syntax

Source Type = Tensor_String_Function (F_Xx = *string_function_xx* | F_Xy = *string_function_xy* | F_Yx = *string_function_yx* | F_Yy = *string_function_yy* | [F_Zx = *string_function_zx*] | [F_Zy = *string_function_zy*] | [F_Xz = *string_function_xz*] | [F_Yz = *string_function_yz*] | [F_Zz = *string_function_zz*]) [Using *element type*] [Averaging Method *{=}* *averaging method*]

Summary

Postprocess using a tensor string function

Parameter	Value	Default
string_function_xx	"string"	–
string_function_xy	"string"	–
string_function_yx	"string"	–
string_function_yy	"string"	–
string_function_zx	"string"	–
string_function_zy	"string"	–
string_function_xz	"string"	–
string_function_yz	"string"	–
string_function_zz	"string"	–
element type	string	–
{=}	{= | are | is}	–
averaging method	string	–

Expression

Syntax

Postprocess Value Of Expression *src name* ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) [Contacting *contacting surface*] [Model *Model Name*] [With Weighting *weight function*] [As *alias name*] [Using *element type*] [Averaging Method *{=} averaging method*]

Summary

Postprocess VALUE of an expression.

Parameter	Value	Default
src name	string	–
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
contacting surface	string	–
Model Name	string	–
weight function	“string”	–
alias name	string	–
element type	string	–
{=}	{= | are | is}	–
averaging method	string	–

Expression

Syntax

Postprocess Value Of Expression *src name* ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over}* Point *Probe Point1 Probe Point2[Probe Point3]* As *alias name* ([With Tolerance *geometric tol*] | [Points_Outside_Tol *{=} {abort | bignum | nan | zero}*] | [Restricted To *Mesh Extent Name. . .*])

Summary

Postprocess an expression

Parameter	Value	Default
src name	string	–
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Probe Point	real1 real2[real3]	–
alias name	string	–
geometric tol	real	–
{=}	{= | are | is}	–
points_outside_tol	{abort | bignum | nan | zero}	–
Mesh Extent Name	string. . .	–

Expression

Syntax

Postprocess Value Of Expression *src name* ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*] | [*{ls}* *{a | b | c}*]) *{@ | at | for | on | over}* Node *Node ID* As *alias name*

Summary

Postprocess an expression

Parameter	Value	Default
src name	string	–
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Node ID	integer	–
alias name	string	–

Expression Shorthand

Syntax

Postprocess *src name* ([*{of} SpeciesName*] | [*{in} MaterialPhaseName*] | [*{ls} {a | b | c}*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) [Contacting *contacting surface*] [Model *Model Name*] [With Weighting *weight function*] [As *alias name*] [Using *element type*] [Averaging Method *{=} averaging method*]

Summary

Postprocess VALUE of an expression.

Parameter	Value	Default
src name	string	–
{of}	{of | species | subindex}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–
{ls}	{levelset_phase | ls}	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
contacting surface	string	–
Model Name	string	–
weight function	“string”	–
alias name	string	–
element type	string	–
{=}	{= | are | is}	–
averaging method	string	–

Field

Syntax

Postprocess Value Of Field *src name {@ | at | for | on | over}* Point *Probe Point1 Probe Point2[Probe Point3]* As *alias name* ([With Tolerance *geometric tol*] | [Points_Outside_Tol *{=} {abort | bignum | nan | zero}*] | [Restricted To *Mesh Extent Name. . .*])

Summary

Postprocess a field

Parameter	Value	Default
src name	string	–
Probe Point	real1 real2[real3]	–
alias name	string	–
geometric tol	real	–
{=}	{= | are | is}	–
points_outside_tol	{abort | bignum | nan | zero}	–
Mesh Extent Name	string. . .	–

Field

Syntax

Postprocess Value Of Field *src name {@ | at | for | on | over}* Node *Node ID* As *alias name*

Summary

Postprocess a field

Parameter	Value	Default
src name	string	–
Node ID	integer	–
alias name	string	–

Function

Syntax

Postprocess Value Of Function *string_function {@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) [Contacting *contacting surface*] [With Weighting *weight function*] As *alias name* [Using *element type*] [Averaging Method *{=} averaging method*]

Summary

Postprocess a scalar string function using a less verbose command

Parameter	Value	Default
string_function	"string"	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
contacting surface	string	–
weight function	"string"	–
alias name	string	–
element type	string	–
{=}	{= | are | is}	–
averaging method	string	–

Function

Syntax

Postprocess Value Of Function *string_function {@ | at | for | on | over}* Point *Probe Point1 Probe Point2[Probe Point3]* As *alias name* ([With Tolerance *geometric tol*] | [Points_Outside_Tol *{=} {abort | bignum | nan | zero}*] | [Restricted To *Mesh Extent Name. . .*])

Summary

Postprocess a scalar string function using a less verbose command

Parameter	Value	Default
string_function	"string"	–
Probe Point	real1 real2[real3]	–
alias name	string	–
geometric tol	real	–
{=}	{= | are | is}	–
points_outside_tol	{abort | bignum | nan | zero}	–
Mesh Extent Name	string. . .	–

Function

Syntax

Postprocess Value Of Function *string_function {@ | at | for | on | over}* Node *Node ID* As *alias name*

Summary

Postprocess a scalar string function using a less verbose command

Parameter	Value	Default
string_function	"string"	–
Node ID	integer	–
alias name	string	–

Scalar_String_Function

Syntax

Postprocess Value Of Scalar_String_Function F = *string_function {@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) [Contacting *contacting surface*] [With Weighting *weight function*] As *alias name* [Using *element type*] [Averaging Method *{=} averaging method*]

Summary

Postprocess using a scalar_string_function

Parameter	Value	Default
string_function	"string"	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
contacting surface	string	–
weight function	"string"	–
alias name	string	–
element type	string	–
{=}	{= | are | is}	–
averaging method	string	–

Scalar_String_Function

Syntax

Postprocess Value Of Scalar_String_Function F = *string_function {@ | at | for | on | over}* Point *Probe Point1 Probe Point2[Probe Point3]* As *alias name* ([With Tolerance *geometric tol*] | [Points_Outside_Tol *{=} {abort | bignum | nan | zero}*] | [Restricted To *Mesh Extent Name. . .*])

Summary

Postprocess using a scalar_string_function

Parameter	Value	Default
string_function	"string"	–
Probe Point	real1 real2[real3]	–
alias name	string	–
geometric tol	real	–
{=}	{= | are | is}	–
points_outside_tol	{abort | bignum | nan | zero}	–
Mesh Extent Name	string. . .	–

Scalar_String_Function

Syntax

Postprocess Value Of Scalar_String_Function F = *string_function {@ | at | for | on | over}* Node *Node ID* As *alias name*

Summary

Postprocess using a scalar_string_function

Parameter	Value	Default
string_function	"string"	–
Node ID	integer	–
alias name	string	–

Tensor_String_Function

Syntax

Postprocess Value Of Tensor_String_Function (F_Xx = *string_function_xx* | F_Xy = *string_function_xy* | F_Yx = *string_function_yx* | F_Yy = *string_function_yy* | [F_Zx = *string_function_zx*] | [F_Zy = *string_function_zy*] | [F_Xz = *string_function_xz*] | [F_Yz = *string_function_yz*] | [F_Zz = *string_function_zz*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) [Contacting *contacting surface*] [With Weighting *weight function*] As *alias name* [Using *element type*] [Averaging Method *{=} averaging method*]

Summary

Postprocess using a tensor_string_function

Parameter	Value	Default
string_function_xx	"string"	–
string_function_xy	"string"	–
string_function_yx	"string"	–
string_function_yy	"string"	–
string_function_zx	"string"	–
string_function_zy	"string"	–
string_function_xz	"string"	–
string_function_yz	"string"	–
string_function_zz	"string"	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
contacting surface	string	–
weight function	"string"	–
alias name	string	–
element type	string	–
{=}	{= | are | is}	–
averaging method	string	–

Tensor_String_Function

Syntax

Postprocess Value Of Tensor_String_Function (F_Xx = *string_function_xx* | F_Xy = *string_function_xy* | F_Yx = *string_function_yx* | F_Yy = *string_function_yy* | [F_Zx = *string_function_zx*] | [F_Zy = *string_function_zy*] | [F_Xz = *string_function_xz*] | [F_Yz = *string_function_yz*] | [F_Zz = *string_function_zz*]) *{@ | at | for | on | over}* Point *Probe Point1 Probe Point2[Probe Point3]* As *alias name* ([With Tolerance *geometric tol*] | [Points_Outside_Tol *{=} {abort | bignum | nan | zero}*] | [Restricted To *Mesh Extent Name. . .*])

Summary

Postprocess using a tensor_string_function

Parameter	Value	Default
string_function_xx	“string”	–
string_function_xy	“string”	–
string_function_yx	“string”	–
string_function_yy	“string”	–
string_function_zx	“string”	–
string_function_zy	“string”	–
string_function_xz	“string”	–
string_function_yz	“string”	–
string_function_zz	“string”	–
Probe Point	real1 real2[real3]	–
alias name	string	–
geometric tol	real	–
{=}	{= | are | is}	–
points_outside_tol	{abort | bignum | nan | zero}	–
Mesh Extent Name	string. . .	–

Tensor_String_Function

Syntax

Postprocess Value Of Tensor_String_Function (F_Xx = *string_function_xx* | F_Xy = *string_function_xy* | F_Yx = *string_function_yx* | F_Yy = *string_function_yy* | [F_Zx = *string_function_zx*] | [F_Zy = *string_function_zy*] | [F_Xz = *string_function_xz*] | [F_Yz = *string_function_yz*] | [F_Zz = *string_function_zz*]) *{@ | at | for | on | over}* Node *Node ID* As *alias name*

Summary

Postprocess using a tensor_string_function

Parameter	Value	Default
string_function_xx	“string”	–
string_function_xy	“string”	–
string_function_yx	“string”	–
string_function_yy	“string”	–
string_function_zx	“string”	–
string_function_zy	“string”	–
string_function_xz	“string”	–
string_function_yz	“string”	–
string_function_zz	“string”	–
Node ID	integer	–
alias name	string	–

Vector_String_Function

Syntax

Postprocess Value Of Vector_String_Function (F_X = *string_function_x* | F_Y = *string_function_y* | [F_Z = *string_function_z*]) *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) [Contacting *contacting surface*] [With Weighting *weight function*] As *alias name* [Using *element type*] [Averaging Method *{=} averaging method*]

Summary

Postprocess using a vector_string_function

Parameter	Value	Default
string_function_x	"string"	–
string_function_y	"string"	–
string_function_z	"string"	–
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
contacting surface	string	–
weight function	"string"	–
alias name	string	–
element type	string	–
{=}	{= | are | is}	–
averaging method	string	–

Vector_String_Function

Syntax

Postprocess Value Of Vector_String_Function (F_X = *string_function_x* | F_Y = *string_function_y* | [F_Z = *string_function_z*]) *{@ | at | for | on | over}* Point *Probe Point1 Probe Point2[Probe Point3]* As *alias name* ([With Tolerance *geometric tol*] | [Points_Outside_Tol *{=} {abort | bignum | nan | zero}*] | [Restricted To *Mesh Extent Name. . .*])

Summary

Postprocess using a vector_string_function

Parameter	Value	Default
string_function_x	"string"	–
string_function_y	"string"	–
string_function_z	"string"	–
Probe Point	real1 real2[real3]	–
alias name	string	–
geometric tol	real	–
{=}	{= | are | is}	–
points_outside_tol	{abort | bignum | nan | zero}	–
Mesh Extent Name	string. . .	–

Vector_String_Function

Syntax

Postprocess Value Of Vector_String_Function (F_X = *string_function_x* | F_Y = *string_function_y* | [F_Z = *string_function_z*]) *{@ | at | for | on | over}* Node *Node ID* As *alias name*

Summary

Postprocess using a vector_string_function

Parameter	Value	Default
string_function_x	"string"	–
string_function_y	"string"	–
string_function_z	"string"	–
Node ID	integer	–
alias name	string	–

Volume

Block Style

Scope

Aria Region, Equation System, Explicit Equation System, Root Finder Equation System

Summary

Specify block-style post processing command

`begin Postprocess Volume`

```
   Location = value ([Touching TouchingMeshExtent]  | [Opposing␣
↪OpposingMeshExtent]) [Contacting contacting surface]

   Output Name = value

end Postprocess Volume
```

Line Commands

Location

Syntax

Location = *value* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) [Contacting *contacting surface*]

Summary

Name of the mesh block or entity to post process.

Parameter	Value	Default
value	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
contacting surface	string	–

Output Name

Syntax

Output Name = *value*

Summary

Name of the output variable or field.

Parameter	Value	Default
value	string	–

Equation

Syntax

Postprocess Volume *{@ | at | for | on | over} Mesh Extent Name* ([Touching *TouchingMeshExtent*] | [Opposing *OpposingMeshExtent*]) [Contacting *contacting surface*] As *alias name*

Summary

Postprocess VOLUME of an expression.

Parameter	Value	Default
Mesh Extent Name	string	–
TouchingMeshExtent	string	–
OpposingMeshExtent	string	–
contacting surface	string	–
alias name	string	–

7.38 Pressurization Model

Pressurization Model

Scope
Aria Region

Summary
Parameters for the pressurization model

```
begin Pressurization Model ModelName

   Coupling Algorithm {=} {coupled | segregated}

   Equation Of State {=} equationOfState...

   Excess Volume {=} excessVolume

   Excess Volume Temperature {=} excessVolumeTemperature

   Excess Volume Weight {=} excessVolumeWeight

   Initial Mixture Molecular Weight {=} initMMW

   Initial Pressure {=} initPressure

   Pressure Unit {=} pressureUnit

   Pressurization Source Blocks {=} sourceBlocks...

   Pressurized Blocks {=} pressurizedBlocks...

   Pressurized Bulk Nodes {=} pressurizedBulkNodes...

   Solver Name {=} solverName
```

```
   Temperature Averaging {=} ventingModel

   Venting Model {=} ventingModel

   Venting Model Property propertyName {=} propertyModelAndParams...

end Pressurization Model ModelName
```

Line Commands

Coupling Algorithm

Syntax

Coupling Algorithm *{=} {coupled | segregated}*

Summary

This command specifies the coupling algorithm (COUPLED or SEGREGATED).

Parameter	Value	Default
{=}	{= | are | is}	–
couplingAlg	{coupled | segregated}	COUPLED

Equation Of State

Syntax

Equation Of State *{=} equationOfState...*

Summary

Specifies the equation of state to use (Ideal_Gas, VanDerWaal, or BKWS). The gas constant is an optional argument (R = value), and defaults to 8314 J/kmol-K if omitted.

Parameter	Value	Default
{=}	{= | are | is}	–
equationOfState	string...	–

Excess Volume

Syntax

Excess Volume *{=} excessVolume*

Summary

Specifies the volume of the excess volume. This volume is assumed to be at the average zone temperature unless you also specify an excess volume temperature.

Parameter	Value	Default
{=}	{= | are | is}	–
excessVolume	"string"	–

Excess Volume Temperature

Syntax

Excess Volume Temperature *{=}* *excessVolumeTemperature*

Summary

Specifies the temperature of the excess volume. If omitted, the excess volume is treated as the same temperature as the main averaged temperature. If provided, you must also provide a weighting factor to use for RHO and RHO_CP temperature averaging schemes.

Parameter	Value	Default
{=}	{= | are | is}	–
excessVolumeTemperature	real	Copied

Excess Volume Weight

Syntax

Excess Volume Weight *{=}* *excessVolumeWeight*

Summary

Specifies the weighting factor to use for the excess volume. For RHO averaging, this will be the density. For rhoCp averaging, this will be rho times Cp. For volume and ideal gas averaging this should not be provided.

Parameter	Value	Default
{=}	{= | are | is}	–
excessVolumeWeight	real	1

Initial Mixture Molecular Weight

Syntax

Initial Mixture Molecular Weight *{=}* *initMMW*

Summary

Specifies the mixture molecular weight of the initial gas present in the pressurization zone

Description

Default is 28.97 g/mol for dry air. This should be specified in the appropriate units for the simulation. If no boundary conditions (e.g. venting or coupling to a

gas transport region of the model) are applied to the pressurization zone then the value does not affect the results of the simulation.

Parameter	Value	Default
{=}	{= | are | is}	–
initMMW	real	28.97

Initial Pressure

Syntax

Initial Pressure *{=}* *initPressure*

Summary

Specifies the initial pressure in the specified pressure units.

Parameter	Value	Default
{=}	{= | are | is}	–
initPressure	real	-1.0

Pressure Unit

Syntax

Pressure Unit *{=}* *pressureUnit*

Summary

This command specifies the units of pressure that is returned. Options are PA, ATM, and PSI (Default is Pa)

Parameter	Value	Default
{=}	{= | are | is}	–
pressureUnit	string	Pa

Pressurization Source Blocks

Syntax

Pressurization Source Blocks *{=}* *sourceBlocks. . .*

Summary

A list of ChemEq blocks that provide the gas source(s) for this pressurization model.

Parameter	Value	Default
{=}	{= | are | is}	–
sourceBlocks	string. . .	–

Pressurized Blocks

Syntax

Pressurized Blocks *{=} pressurizedBlocks. . .*

Summary

A list of blocks that the pressurization occurs in.

Parameter	Value	Default
{=}	{= | are | is}	–
pressurizedBlocks	string. . .	–

Pressurized Bulk Nodes

Syntax

Pressurized Bulk Nodes *{=} pressurizedBulkNodes. . .*

Summary

A list of bulk nodes that the pressurization occurs in.

Parameter	Value	Default
{=}	{= | are | is}	–
pressurizedBulkNodes	string. . .	–

Solver Name

Syntax

Solver Name *{=} solverName*

Summary

This command specifies the name of the ODE solver block to use for pressurization.

Parameter	Value	Default
{=}	{= | are | is}	–
solverName	string	NONE

Temperature Averaging

Syntax

Temperature Averaging *{=} ventingModel*

Summary

This command specifies the temperature averaging model to use (VOLUME, IDEAL_GAS, RHO, or RHOCP)

Parameter	Value	Default
{=}	{= | are | is}	–
ventingModel	string	IDEAL_GAS

Venting Model

Syntax

Venting Model *{=}* *ventingModel*

Summary

This command specifies the venting model to use (OPEN, CLOSED, or VENTED)

Parameter	Value	Default
{=}	{= | are | is}	–
ventingModel	string	–

Venting Model Property

Syntax

Venting Model Property *propertyName* *{=}* *propertyModelAndParams. . .*

Summary

Define a material property associated with the venting model. Required if the venting model is VENTED. At a minimum a model for the VENTING_VOLUMETRIC_FLOW_RATE must be provided, and depending on the choice of that model additional properties may be required.

Parameter	Value	Default
propertyName	string	–
{=}	{= | are | is}	–
propertyModelAndParams	string. . .	–

7.39 Restart

Restart Data

Scope

Aria Region, Input_Output Region

Summary

Describes the data required to output and input restart data for the enclosing region.

```
begin Restart Data Label

   Additional Steps {=} List_of_steps...

   Additional Times {=} List_of_times...
```

```
  At Step n {increment | interval} {=} m

  At Time Dt1 {increment | interval} {=} Dt2

  At Wall Time Dt1 {increment | interval} {=} Dt2

  Component Separator Character {=} Separator

  Cycle Count {=} Count

  Database Name {=} StreamName

  Database Type {=} {catalyst | catalyst_exodus | cgns | dof | dof_
↪exodus | exodus | exodusii | exonull | generated | genesis | null |␣
↪parallel_exodus | textmesh}

  Debug Dump

  Decomposition Method {=} {block | cyclic | external | geom_kway | hsfc␣
↪| kway | kway_geom | linear | map | metis_sfc | random | rcb | rib |␣
↪variable}

  Exists {=} {abort | add_suffix | append | overwrite}

  File Cycle Count {=} Count

  Flush Interval {=} Option

  Input Database Name {=} StreamName

  Optional

  Output Database Name {=} StreamName

  Output On Signal {=} {sigabrt | sigalrm | sigfpe | sighup | sigill |␣
↪sigint | sigkill | sigpipe | sigquit | sigsegv | sigterm | sigusr1 |␣
↪sigusr2}

  Output Restart State {=} {off | on}

  Overlay Count {=} Count

  Overwrite {=} {false | no | off | on | true | yes}
```

```
   Property PropertyName {=} PropertyValue

   Restart {=} {auto}

   Restart Time {=} Time

   Start Time {=} Start_time

   Synchronize Output

   Shift To Start Time

   Termination Time {=} Final_time

   Timestep Adjustment Interval {=} Nsteps

   Use Dynamic Topology Io

   Use External Control

   Use Output Scheduler Timer_name

end Restart Data Label
```

Line Commands

Additional Steps

Syntax

Additional Steps *{=}* *List_of_steps. . .*

Summary

Additional simulation steps when output should occur.

Parameter	Value	Default
{=}	{= | are | is}	–
List_of_steps	integer. . .	–

Additional Times

Syntax

Additional Times *{=}* *List_of_times. . .*

Summary

Additional simulation times when output should occur.

Parameter	Value	Default
{=}	{= | are | is}	–
List_of_times	real. . .	–

At Step

Syntax

At Step *n {increment | interval} {=} m*

Summary

Specify an output interval in terms of the internal iteration step count. The first step specifies the step count at the beginning of this interval and the second step specifies the output frequency to be used within this interval.

Parameter	Value	Default
n	integer	–
Option	{increment | interval}	–
{=}	{= | are | is}	–
m	integer	–

At Time

Syntax

At Time *Dt1 {increment | interval} {=} Dt2*

Summary

Specify an output interval in terms of the internal simulation time. The first time specifies the time at the beginning of this time interval and the second time specifies the output frequency to be used within this interval.

Parameter	Value	Default
Dt1	real	–
Option	{increment | interval}	–
{=}	{= | are | is}	–
Dt2	real	–

At Wall Time

Syntax

At Wall Time *Dt1 {increment | interval} {=} Dt2*

Summary

Write a restart file at a specific wall time since the start of the run. Time string format allows s, m, h, d for seconds, minutes, hours, days

Parameter	Value	Default
Dt1	string	–
Option	{increment | interval}	–
{=}	{= | are | is}	–
Dt2	string	–

Component Separator Character

Syntax

Component Separator Character *{=} Separator*

Summary

The separator is the single character used to separate the output variable basename (e.g. "stress") from the suffices (e.g. "xx", "yy") when displaying the names of the individual variable components. For example, the default separator is "_", which results in names similar to "stress_xx", "stress_yy", . . . "stress_zx". To eliminate the separator, specify an empty string ("") or NONE.

Parameter	Value	Default
{=}	{= | is}	–
Separator	string	–

Cycle Count

Syntax

Cycle Count *{=} Count*

Summary

Specify the number of restart steps which will be written to the restart database before previously written steps are overwritten. For example, if the cycle count is 5 and restart is written every 0.1 seconds, the restart system will write 0.1, 0.2, 0.3, 0.4, 0.5 to the database. It will then overwrite the first step with data from time 0.6, the second with time 0.7. At time 0.8, the database would contain data at times 0.6, 0.7, 0.8, 0.4, 0.5. Note that time will not necessarily be monotonically increasing on a database that specifies the cycle count.

Parameter	Value	Default
{=}	{= | are | is}	–
Count	integer	–

Database Name

Syntax

Database Name *{=} StreamName*

Summary

The database containing the input and/or output restart data. If this analysis is being restarted, restart data will be read from this file. If the analysis is writing restart data, the data will be written to this file. It will be overwritten if it exists (after being read if applicable). If the filename begins with the '/' character, it is an absolute path; otherwise, the path to the current directory will be prepended to the name. See also the 'Input Database' and 'Output Database' commands.

Parameter	Value	Default
{=}	{= | are | is}	–
StreamName	string	–

Database Type

Syntax

Database Type *{=} {catalyst | catalyst_exodus | cgns | dof | dof_exodus | exodus | exodusii | exonull | generated | genesis | null | parallel_exodus | textmesh}*

Summary

The database type/format used for the restart file.

Parameter	Value	Default
{=}	{= | are | is}	–
Database-Types	{catalyst | catalyst_exodus | cgns | dof | dof_exodus | exodus | exodusii | exonull | generated | genesis | null | parallel_exodus | textmesh}	–

Debug Dump

Syntax

Debug Dump

Summary

Specify whether the restart system will write the restart data immediately after reading the restart data if the run is restarting. The output data can be compared with the restart input data to determine whether they match.

Decomposition Method

Syntax

Decomposition Method *{=} {block | cyclic | external | geom_kway | hsfc | kway | kway_geom | linear | map | metis_sfc | random | rcb | rib | variable}*

Summary

The decomposition algorithm to be used to partition elements to each processor in a parallel run.

Parameter	Value	Default
{=}	{= | are | is}	–
Method	{block | cyclic | external | geom_kway | hsfc | kway | kway_geom | linear | map | metis_sfc | random | rcb | rib | variable}	–

Exists

Syntax

Exists *{=} {abort | add_suffix | append | overwrite}*

Summary

Specify the behavior when creating this database and there is an existing file with the same name. The default behavior is "OVERWRITE" which deletes the existing file and creates a new file of the same name. "APPEND" will (if possible) append the new data to the end of the existing file. "ABORT" will print an error message and end the analysis. "ADD_SUFFIX" will add a suffix to the file name and output to that file.

Parameter	Value	Default
{=}	{= | is}	–
Option2	{abort | add_suffix | append | overwrite}	–

File Cycle Count

Syntax

File Cycle Count *{=} Count*

Summary

Each restart dump will be written to a separate file suffixed with A,B, . . . The count specifies how many separate files are used before the cycle repeats. For example, if "FILE CYCLE COUNT = 3" is specified, the restart dumps would be written to file-A.rs, file-B.rs, file-C.rs, file-A.rs, . . . The maximum value for the cycle count is 26.

Parameter	Value	Default
{=}	{= | are | is}	–
Count	integer	–

Flush Interval

Syntax

Flush Interval *{=} Option*

Summary

The minimum time interval (in seconds) at which output will be explicitly flushed to disk. The default is 10 seconds.

Parameter	Value	Default
{=}	{= | are | is}	–
Option	integer	10

Input Database Name

Syntax

Input Database Name *{=}* *StreamName*

Summary

The database containing the input restart data. If this analysis is being restarted, restart data will be read from this file. See also the 'Database' and 'Output Database' commands.

Parameter	Value	Default
{=}	{= | are | is}	–
StreamName	string	–

Optional

Syntax

Optional

Summary

The database will be read if it exists, but it is not an error if there is no restart database to read for this region during a restarted analysis.

Output Database Name

Syntax

Output Database Name *{=}* *StreamName*

Summary

The database containing the output restart data. If the analysis is writing restart data, the data will be written to this file. It will be overwritten if it exists. See also the 'Database' and 'Input Database' commands.

Parameter	Value	Default
{=}	{= | are | is}	–
StreamName	string	–

Output On Signal

Syntax

Output On Signal *{=} {sigabrt | sigalrm | sigfpe | sighup | sigill | sigint | sigkill | sigpipe | sigquit | sigsegv | sigterm | sigusr1 | sigusr2}*

Summary

When the specified signal is raised, the output stream associated with this block will be output.

Parameter	Value	Default
{=}	{= | are | is}	–
Signals	{sigabrt | sigalrm | sigfpe | sighup | sigill | sigint | sigkill | sigpipe | sigquit | sigsegv | sigterm | sigusr1 | sigusr2}	–

Output Restart State

Syntax

Output Restart State *{=} {off | on}*

Summary

Outputs the restarted state to the new restarted results file

Description

NOTE: This command must be placed at the Sierra scope of the input file. Allows the analyst to visualize the restarted state for debugging

Parameter	Value	Default
{=}	{= | are | is}	–
Option	{off | on}	–

Overlay Count

Syntax

Overlay Count *{=} Count*

Summary

Specify the number of restart outputs which will be overlaid on top of the last written step. For example, if restarts are being output every 0.1 seconds and the overlay count is specified as 2, then restart will write times 0.1 to step 1 of the database. It will then write 0.2 and 0.3 also to step 1. It will then increment the database step and write 0.4 to step 2; overlay 0.5 and 0.6 on step 2. . . At the end of the analysis, assuming it runs to completion, the database would have times 0.3, 0.6, 0.9, . . . However, if there were a problem during the analysis, the last step on the database would contain an intermediate step.

Parameter	Value	Default
{=}	{= | are | is}	–
Count	integer	–

Overwrite

Syntax

Overwrite *{=} {false | no | off | on | true | yes}*

Summary

(DEPRECATED, Use EXISTS) Specify whether the restart database should be overwritten if it exists. The default behavior is to overwrite unless this command is specified in the restart block and either off, false, or no is specified.

Parameter	Value	Default
{=}	{= | is}	–
Option2	{false | no | off | on | true | yes}	–

Property

Syntax

Property *PropertyName {=} PropertyValue*

Summary

Define a database property named “PropertyName” with the value “PropertyValue”. If PropertyValue consists of all digits, it will define an integer property. If PropertyValue is “true” or “yes” or “false” or “no”, it will define a logical property; otherwise it will define a string property. If PropertyName consists of multiple strings, they will be concatenated together with “_” separating the individual words. Supported properties are typically database dependent; Current properties are:

- COMPRESSION_LEVEL = [0..9]
- COMPRESSION_SHUFFLE = true|false|on|off
- FILE_TYPE = netcdf4 (forces use of netcdf-4 hdf5-based file)
- INTEGER_SIZE_DB = 4|8
- INTEGER_SIZE_API = 4|8
- LOGGING = true|false|on|off
- MAX_NAME_LENGTH = value

Parameter	Value	Default
PropertyName	string	–
{=}	{= | are | is}	–
PropertyValue	string	–

Restart

Syntax

Restart *{=}* *{auto}*

Summary

Specify automatic restart file read.

Description

NOTE: This command must be placed at the Sierra scope of the input file.

Specify that the analysis should be restarted from the last common time on all restart databases for each Region in the analysis. In addition to this line command, each Region in the analysis (strictly, only the region(s) that will be restarted) must have a restart block specifying the database to read the restart state data.

By default, use of this command will not cause output files (e.g., results, history, heartbeat, restart) to be overwritten. Instead output files will be written with the same basename and the suffix `-s000*`. Common visualization packages are written to handle this file organization gracefully in order for the user to view all results seamlessly.

Parameter	Value	Default
{=}	{= | are | is}	–
{auto}	{auto | automatic}	–

Restart Time

Syntax

Restart Time *{=}* *Time*

Summary

Specify restart file read at a specified time.

Description

NOTE: This command must be placed at the Sierra scope of the input file.

Specify the time that the analysis will be restarted. In addition to this line command, each Region in the analysis (strictly, only the region(s) that will be restarted) must have a restart block specifying the database to read the restart state data. The restart 'time' must be greater than zero and less than or equal to the termination time.

By default, use of this command will cause previous output files (e.g., results, history, heartbeat, restart) to be overwritten. If this command is chosen, the onus is placed on the user to ensure that previous output files are not overwritten.

Parameter	Value	Default
{=}	{= | are | is}	–
Time	real	–

Start Time

Syntax

Start Time *{=}* *Start_time*

Summary

Specify the time to start outputting results from this output request block. This time overrides all 'at time' and 'at step' specifications.

Parameter	Value	Default
{=}	{= | are | is}	–
Start_time	real	–

Synchronize Output

Syntax

Synchronize Output

Summary

In an analysis with multiple regions, it is sometimes desirable to synchronize the output of results data between the regions. This can be done by adding the SYNCHRONIZE OUTPUT command line to the results output block. If a results block has this set, then it will write output whenever a previous region writes output. The ordering of regions is based on the order in the input file, algorithmic considerations, or by solution control specifications.

Although the USE OUTPUT SCHEDULER command line can also synchronize output between regions, the SYNCHRONIZE OUTPUT command line will synchronize the output with regions where the output frequency is not under the direct control of the Sierra IO system. Examples of this are typically coupled applications where one or more of the codes are not Sierra-based applications such as Alegra and CTH. A results block with SYNCHRONIZE OUTPUT specified will also synchronize its output with the output of the external code.

The SYNCHRONIZE OUTPUT command can be used with other output scheduling commands such as time-based or step-based output specifications.

Shift To Start Time

Syntax

Shift To Start Time

Summary

The shift to start time option allows a user to shift the restart time to the start time of the current region. An example use case would be if a restart time of 0.5 is specified, but the user would like to start the simulation at time 1.0.

Termination Time

Syntax

Termination Time *{=}* *Final_time*

Summary

Specify the time to stop outputting results from this output request block.

Parameter	Value	Default
{=}	{= | are | is}	–
Final_time	real	–

Timestep Adjustment Interval

Syntax

Timestep Adjustment Interval *{=}* *Nsteps*

Summary

Specify the number of steps to 'look ahead' and adjust the timestep to ensure that the specified output times or simulation end time will be hit 'exactly'.

Parameter	Value	Default
{=}	{= | are | is}	–
Nsteps	integer	–

Use Dynamic Topology Io

Syntax

Use Dynamic Topology Io

Summary

Specify that the app use IO for dynamic topology modifications where the output files are stored in a single database. Legacy file format for dynamically changing topology results in the creation of multiple files for each output on a mesh modification. This option leverages the ability of netCDF to create mesh groups within a single database and concatenate all mesh files into one. The names of each mesh group are of the form IOSS_MESH_GROUP-??? where ??? is the 1-based output index 1, 2, . . . , 10, . . ., 100, . . . Please note that netCDF has a current limit of 65,536 groups

Use External Control

Syntax

Use External Control

Summary

Do not schedule any results output but instead depend on an externally coupled application to explicitly call for results output.

Use Output Scheduler

Syntax

Use Output Scheduler *Timer_name*

Summary

Associates a predefined output scheduler with this output block (results, restart, heartbeat, or history).

Parameter	Value	Default
Timer_name	string	–

Restart Time

Syntax

Restart Time *{=}* *Time*

Summary

Specify restart file read at a specified time.

Description

NOTE: This command must be placed at the Sierra scope of the input file.

Specify the time that the analysis will be restarted. In addition to this line command, each Region in the analysis (strictly, only the region(s) that will be restarted) must have a restart block specifying the database to read the restart state data. The restart 'time' must be greater than zero and less than or equal to the termination time.

By default, use of this command will cause previous output files (e.g., results, history, heartbeat, restart) to be overwritten. If this command is chosen, the onus is placed on the user to ensure that previous output files are not overwritten.

Parameter	Value	Default
{=}	{= | are | is}	–
Time	real	–

7.40 Solution Options

Solution Options

Scope

Aria Region

Summary

Specify information regarding the governing equations to be solved.

```
begin Solution Options OptionsName

   Apply Flux Limiter Stabilization

   Check Matrix For Discrete Maximum Principle

   Face Stabilization Multiplier {=} Multiplier

   Force Non Tale {=} {false | no | off | on | true | yes}

   Free Stream Reynolds Number {=} input_Re

   Ignore Coordinate Displacements {=} {false | no | off | on | true |␣
↪yes}

   Maximum Temperature Allowed From Temperature Extraction {=} MaxTemp

   Maximum Wall Time {=} WallTime Hours

   Minimum Temperature Allowed From Temperature Extraction {=} MinTemp

   Omit Enthalpy Adjustment After Temperature Clipping

   Omit Finite Difference Sensitivities For Cantera Properties

   Use Inverse Density Continuity Scaling {=} {false | no | off | on |␣
↪true | yes}

   begin Cvfem Algorithm Specification PStabName
   end

   begin Edc Model Specification EdcSpecName
   end

   begin Hdiff Model Specification HdiffOptionsName
```

```
  end

  begin Porous Flow Options blockName
  end

  begin Species Options blockName
  end

  begin Turbulence Model Specification TurbSpecName
  end

end Solution Options OptionsName
```

Line Commands

Apply Flux Limiter Stabilization

Syntax

Apply Flux Limiter Stabilization

Summary

Enables flux limiter stabilization for diffusion.

Description

This activates flux limiter stabilization for diffusion terms to prevent non-physical temperatures on meshes that don't satisfy the requirements for a discrete maximum principle, or for anisotropic material properties.

Check Matrix For Discrete Maximum Principle

Syntax

Check Matrix For Discrete Maximum Principle

Summary

Examine rows of the system matrix before boundary conditions are applied and check if the diagonal entries are non-negative, the off-diagonal entries are non-positive, the row-sum is non-negative. If these conditions fail, it is possible the solution will contain non-physical values, and the discrete maximum principle is not satisfied by the matrix. If any of these conditions fail, a warning is printed to the log file. More output can be obtained with turning on debug logging.

Face Stabilization Multiplier

Syntax

Face Stabilization Multiplier *{=}* *Multiplier*

Summary

Multiplier for face stabilization term.

Description

This value specifies the multiplier applied to the stabilization terms activated by the GRAD_JUMP_PENALTY or DMP stabilization terms on the specified equation.

Parameter	Value	Default
{=}	{= | are | is}	–
Multiplier	real	1.0

Force Non Tale

Syntax

Force Non Tale *{=} {false | no | off | on | true | yes}*

Summary

Enable (true) to force simulation not to use TALE.

Parameter	Value	Default
{=}	{= | are | is}	–
Value	{false | no | off | on | true | yes}	FALSE

Free Stream Reynolds Number

Syntax

Free Stream Reynolds Number *{=} input_Re*

Summary

Input Reynolds number is used for solving nondimensional viscous problems using the GasDyn equations.

Parameter	Value	Default
{=}	{= | are | is}	–
input_Re	real	–

Ignore Coordinate Displacements

Syntax

Ignore Coordinate Displacements *{=} {false | no | off | on | true | yes}*

Summary

Enable (true) or disable (false) displacing the physical coordinates by either MESH_DISPLACEMENTS or SOLID_DISPLACEMENTS. By default, this is false, and physical_coordinates are displaced. However, this can be disabled by setting this to true, and physical_coordinates will be equal to model_coordinates.

Parameter	Value	Default
{=}	{= | are | is}	–
Value	{false | no | off | on | true | yes}	FALSE

Maximum Temperature Allowed From Temperature Extraction

Syntax

Maximum Temperature Allowed From Temperature Extraction *{=}* *MaxTemp*

Summary

Specify a maximum cutoff temperature for extraction from enthalpy and composition

Description

This option specifies the maximum temperature to be allowed to be extracted from enthalpy, given the mixture composition. If a temperature is computed that is greater than this value, the temperature is reset to equal the maximum allowed value.

Parameter	Value	Default
{=}	{= | are | is}	–
MaxTemp	real	2400.0 K

Maximum Wall Time

Syntax

Maximum Wall Time *{=}* *WallTime* Hours

Summary

Specify a maximum wall time to let the simulation end gracefully and output before slurm kills it.

Parameter	Value	Default
{=}	{= | are | is}	–
WallTime	real	Infinite

Minimum Temperature Allowed From Temperature Extraction

Syntax

Minimum Temperature Allowed From Temperature Extraction *{=}* *MinTemp*

Summary

Specify a minimum cutoff temperature for extraction from enthalpy and composition

Description

This option specifies the minimum temperature to be allowed to be extracted

from enthalpy, given the mixture composition. If a temperature is computed that is less than this value, the temperature is reset to equal the minimum allowed value.

Parameter	Value	Default
{=}	{= | are | is}	–
MinTemp	real	250.0 K

Omit Enthalpy Adjustment After Temperature Clipping

Syntax

Omit Enthalpy Adjustment After Temperature Clipping

Summary

Do not adjust enthalpy to be consistent with clipped temperatures

Description

When temperature extraction fails to produce a viable result, the temperature will usually be clipped at either the upper or lower limit temperature. By default, the enthalpy will then be adjusted up or down to match the clipped temperature so that the thermochemical state is internally consistent. This omits the enthalpy adjustment, leaving clipped nodes in an inconsistent state.

Omit Finite Difference Sensitivities For Cantera Properties

Syntax

Omit Finite Difference Sensitivities For Cantera Properties

Summary

Do not populate any Cantera property sensitivities via finite difference

Use Inverse Density Continuity Scaling

Syntax

Use Inverse Density Continuity Scaling *{=} {false | no | off | on | true | yes}*

Summary

Enable (true) or disable (false) the inverse element scaling of the continuity equation.

Parameter	Value	Default
{=}	{= | are | is}	–
Value	{false | no | off | on | true | yes}	TRUE

Cvfem Algorithm Specification

Summary

Specify CVFEM algorithmic modeling options.

```
begin Cvfem Algorithm Specification PStabName

  Activate Acoustic Compressibility Algorithm

  Activate Ausm Plus Scheme

  Activate Edge Based Diffusion Operator [ With {minimum_nonorthogonal_
↪correction | orthogonal_nonorthogonal_correction | overrelaxed_
↪nonorthogonal_correction}  ]

  Activate Edge Based Fourth Order Advection

  Activate Kexact Muscl Scheme For Equation EquationString ([{of}␣
↪SpeciesName]  | [{in} MaterialPhaseName]) [ Using K {=} kappa  ]

  Activate Mixture Fraction Clipping Utility At {begin_nonlinear_solve␣
↪| end_nonlinear_solve | manually_run | post_iterate | post_linear_
↪solve | post_nonlinear_solve | pre_iterate | pre_linear_solve | pre_
↪nonlinear_solve | run_initially | run_once | run_post_linear_system_
↪initialization}

  Activate Muscl Scheme For Equation EquationString ([{of} SpeciesName] ␣
↪| [{in} MaterialPhaseName])

  Activate Pressure Projection Algorithm

  Activate Projected Stress Stabilization

  Activate Scv Nodal Gradient

  Activate Shakib Scaling

  Activate Vrtm In Mass Flux Vector

  Ausm Alpha {=} alpha

  Ausm Beta {=} beta

  Clip Cvfem Level Set Dof
```

First Order Upwind Factor {=} RealValue For Equation EquationString ([
↪{of} SpeciesName] | [{in} MaterialPhaseName])

Flux Scheme {=} {ausm | ausm_plus | roe | van_leer}

Freeze Muscl Limiter At Global Step {=} freezeStep

Hybrid Upwind Factor {=} RealValue For Equation EquationString ([{of}␣
↪SpeciesName] | [{in} MaterialPhaseName])

Interpolate Density And Velocity Separately In Mass Flux Vector

Lag Nodal Pressure Gradient In Mass Flux Vector Expression

Lag Nodal Tau In Mass Flux Vector Expression

Muscl Limiter {=} {muscl_minmod | muscl_none | muscl_superbee |␣
↪muscl_van_albada | muscl_van_leer} For Equation EquationString ([{of}
↪ SpeciesName] | [{in} MaterialPhaseName])

Omit Diffusion Term From Sucv Tau

Omit Disting Bc Ip Sensitivities

Omit Dt Term From Sucv Tau

Pressure Stabilization Characteristic Length {=} CharLength

Pressure Stabilization Order {=} {fourth_order | second_order | zeroth_
↪order}

Pressure Stabilization Parameter Scaling {=} TauScaling

Pressure Stabilization Scaling {=} {characteristic | constant | shakib␣
↪| stabilized | time_step} [With Value {=} ConstantTau]

Roe Entropy Fix C {=} epsc

Roe Entropy Fix U {=} epsu

Upwind Method {=} {fourth | gupw | muscl | sucv | upw} For Equation␣
↪EquationString ([{of} SpeciesName] | [{in} MaterialPhaseName])

Use Approximate Roe Sensitivities

```
   Use Opposing Subface In Open Mass Flux

   Use Specified Pressure In Open Mass Flux
```

end Cvfem Algorithm Specification *PStabName*

Line Commands

Activate Acoustic Compressibility Algorithm

Syntax

Activate Acoustic Compressibility Algorithm

Summary

Variable thermodynamic pressure to allow for closed system pressurization

Activate Ausm Plus Scheme

Syntax

Activate Ausm Plus Scheme

Summary

Activate AUSM plus with standard values for alpha and beta

Description

Activate AUSM plus scheme.

Activate Edge Based Diffusion Operator

Syntax

Activate Edge Based Diffusion Operator [With *{minimum_nonorthogonal_correction | orthogonal_nonorthogonal_correction | overrelaxed_nonorthogonal_correction}*]

Summary

Activate edge-based scheme for diffusion operator.

Description

Only implemented now for tke and sdr - interior only

Activate Edge Based Fourth Order Advection

Syntax

Activate Edge Based Fourth Order Advection

Summary

Activate edge-based advection for fourth order LES scheme

Description

Activate edge-based advection for fourth order LES scheme

Activate Kexact Muscl Scheme For Equation

Syntax

Activate Kexact Muscl Scheme For Equation *EquationString* ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*]) [Using K *{=}* *kappa*]

Summary

Use kexact MUSCL variable extrapolation for the specified equation

Description

Use kexact MUSCL variable extrapolation for the specified equation. The projected gradients for the necessary variables must be computed using a cvfem_lumped_muscl_projection equation.

Parameter	Value	Default
EquationString	string	–
{of}	{of | species}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–

Activate Mixture Fraction Clipping Utility At

Syntax

Activate Mixture Fraction Clipping Utility At *{begin_nonlinear_solve | end_nonlinear_solve | manually_run | post_iterate | post_linear_solve | post_nonlinear_solve | pre_iterate | pre_linear_solve | pre_nonlinear_solve | run_initially | run_once | run_post_linear_system_initialization}*

Summary

Activate clipping utility for mixture fraction

Description

Sometimes it helps to clip the mixture fraction. This is a utility that provides this support.

Parameter	Value	Default
UtilClipLocation	{begin_nonlinear_solve | end_nonlinear_solve | manually_run | post_iterate | post_linear_solve | post_nonlinear_solve | pre_iterate | pre_linear_solve | pre_nonlinear_solve | run_initially | run_once | run_post_linear_system_initialization}	–

Activate Muscl Scheme For Equation

Syntax

Activate Muscl Scheme For Equation *EquationString* ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*])

Summary

Use MUSCL variable extrapolation for the specified equation

Description

Use MUSCL variable extrapolation for the specified equation. The projected gradients for the necessary variables must be computed using a cvfem_lumped_muscl_projection equation.

Parameter	Value	Default
EquationString	string	–
{of}	{of | species}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–

Activate Pressure Projection Algorithm

Syntax

Activate Pressure Projection Algorithm

Summary

Activate PP alg.

Description

Rather than fully coupling uvwp, activate a pressure projection algorithm. This will require creating two EQ systems: uvw and p. However, once this is created, the code will handle supplemental utilities. This option can only be used in the context of a momentum lumped mass matrix

Activate Projected Stress Stabilization

Syntax

Activate Projected Stress Stabilization

Summary

Activate full momentum stress term for residual stabilization

Description

Activate full momentum stress term for residual stabilization

Activate Scv Nodal Gradient

Syntax

Activate Scv Nodal Gradient

Summary

Use scv nodal gradient for MUSCL and edge-based diffusion terms

Description

Use scv nodal gradient for MUSCL and edge-based diffusion terms; default is to use nodal grad based on Green Gauss over the duel mesh with edge based integration points.

Activate Shakib Scaling

Syntax

Activate Shakib Scaling

Summary

Specify Shakib scaling; temp line command

Activate Vrtm In Mass Flux Vector

Syntax

Activate Vrtm In Mass Flux Vector

Summary

Activate velocity relative to mesh in mass flux vector

Description

In some cases, the volume may be constant, although there is mesh motion. This is the case in wind energy sliding mesh applications. This algorithm pays not attention to GCL and assumes that the time rate of volume is zero. Again, this is fine in solid rotation.

Ausm Alpha

Syntax

Ausm Alpha *{=}* *alpha*

Summary

Set constant alpha in Mach splitting function.

Description

Default is 0, unless ausm+ is chosen, then the default is 3/16.

Parameter	Value	Default
{=}	{= | are | is}	–
alpha	real	0.0

Ausm Beta

Syntax

Ausm Beta *{=}* *beta*

Summary

Set constant beta in Mach splitting function.

Description

Default is 0, unless ausm+ is chosen, then the default is 1/8..

Parameter	Value	Default
{=}	{= | are | is}	–
beta	real	0.0

Clip Cvfem Level Set Dof

Syntax

Clip Cvfem Level Set Dof

Summary

Clip level set; only germane for conserved level set formulation that has bounds between zero and unity

First Order Upwind Factor

Syntax

First Order Upwind Factor *{=}* *RealValue* For Equation *EquationString* ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*])

Summary

First-order upwind factor, $0 < x \leq 1$.

Description

This value specifies the explicit upwind blending between pure upwind and the chosen convection operator, e.g., UPW*(firstOrderUpwind) + (1-firstOrderUpwind)*(blendedUpwindCentral).

where UPW is the pure first order upwind value and blendedUpwindCentral is a blend between the selected upwind method and central difference operator based on the local cell Peclet number (see Hybrid Upwind Factor line command).

For now, the blendedUpwindCentral is pure central.

Parameter	Value	Default
{=}	{= | are | is}	–
RealValue	real	1.0
EquationString	string	–
{of}	{of | species}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–

Flux Scheme

Syntax

Flux Scheme *{=} {ausm | ausm_plus | roe | van_leer}*

Summary

Set flux scheme used by all equations in the high-Mach CVFEM formulation.

Description

Set flux scheme used by all equations in the high-Mach CVFEM formulation.

Parameter	Value	Default
{=}	{= | are | is}	–
FluxScheme	{ausm | ausm_plus | roe | van_leer}	–

Freeze Muscl Limiter At Global Step

Syntax

Freeze Muscl Limiter At Global Step *{=} freezeStep*

Summary

Freeze MUSCL limiter at a given global step

Description

Freeze the MUSCL limiter

Parameter	Value	Default
{=}	{= | are | is}	–
freezeStep	real	–

Hybrid Upwind Factor

Syntax

Hybrid Upwind Factor *{=} RealValue* For Equation *EquationString* ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*])

Summary

Hybrid upwind factor dials in blending between user specified operator and central.

Description

Allows blending of the Peclet factor. A value of zero produces pure central. Higher values blend more user specified upwind (currently, pure first order upwind).

Parameter	Value	Default
{=}	{= | are | is}	–
RealValue	real	1.0
EquationString	string	–
{of}	{of | species}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–

Interpolate Density And Velocity Separately In Mass Flux Vector

Syntax

Interpolate Density And Velocity Separately In Mass Flux Vector

Summary

use (rho)ip*(uj)ip rather than default (rho*uj)ip

Lag Nodal Pressure Gradient In Mass Flux Vector Expression

Syntax

Lag Nodal Pressure Gradient In Mass Flux Vector Expression

Summary

Compute the nodal pressure gradient calculation.

Description

Lag the nodal pressure gradient assembly for use in the stabilized mass flux vector expression. This command will place the calculation at the top of the non-linear iteration. In general, it will not be wise to use this option for steady simulations as there is one nonlinear loop.

Lag Nodal Tau In Mass Flux Vector Expression

Syntax

Lag Nodal Tau In Mass Flux Vector Expression

Summary

Compute the nodal tau calculation.

Description

Lag the nodal tau assembly for use in the stabilized mass flux vector expression. This command will place the calculation at the top of the non-linear iteration. In general, it will not be wise to use this option for steady simulations as there is one nonlinear loop.

Muscl Limiter

Syntax

Muscl Limiter *{=}* *{muscl_minmod* | *muscl_none* | *muscl_superbee* |

muscl_van_albada | muscl_van_leer} For Equation *EquationString* ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*])

Summary

Specify the MUSCL limiter method.

Description

TVD limiter.

Parameter	Value	Default
{=}	{= | are | is}	–
MusclLimiter	{muscl_minmod | muscl_none | muscl_superbee | muscl_van_albada | muscl_van_leer}	–
EquationString	string	–
{of}	{of | species}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–

Omit Diffusion Term From Sucv Tau

Syntax

Omit Diffusion Term From Sucv Tau

Summary

Do not use timestep in the expression for the SUCV stabilization coefficient

Omit Disting Bc Ip Sensitivities

Syntax

Omit Disting Bc Ip Sensitivities

Summary

Drop distinguishing bc sensitivities for ip values.

Description

Drop distinguishing bc sensitivities for ip values.

Omit Dt Term From Sucv Tau

Syntax

Omit Dt Term From Sucv Tau

Summary

Do not use timestep in the expression for the SUCV stabilization coefficient

Pressure Stabilization Characteristic Length

Syntax

Pressure Stabilization Characteristic Length *{=}* *CharLength*

Summary

Specify a constant length scale to compute tau; only applicable for characteristic scaling

Parameter	Value	Default
{=}	{= | are | is}	–
CharLength	real	–

Pressure Stabilization Order

Syntax

Pressure Stabilization Order *{=}* *{fourth_order | second_order | zeroth_order}*

Summary

Order of pressure stabilization

Parameter	Value	Default
{=}	{= | are | is}	–
Cvfempstaborder	{fourth_order | second_order | zeroth_order}	–

Pressure Stabilization Parameter Scaling

Syntax

Pressure Stabilization Parameter Scaling *{=}* *TauScaling*

Summary

Scaling of tau

Parameter	Value	Default
{=}	{= | are | is}	–
TauScaling	real	–

Pressure Stabilization Scaling

Syntax

Pressure Stabilization Scaling *{=}* *{characteristic | constant | shakib | stabilized | time_step}* [With Value *{=}* *ConstantTau*]

Summary

Scaling coefficient for pressure stabilization

Parameter	Value	Default
{=}	{= | are | is}	–
CvfemPStabScaling	{characteristic | constant | shakib | stabilized | time_step}	–

Roe Entropy Fix C

Syntax

Roe Entropy Fix C *{=}* *epsc*

Summary

Small constant for entropy fix (modification of |U +/- c| eigenvalues in Roe scheme)

Parameter	Value	Default
{=}	{= | are | is}	–
epsc	real	0.1

Roe Entropy Fix U

Syntax

Roe Entropy Fix U *{=}* *epsu*

Summary

Small constant for entropy fix (modification of |U| eigenvalues in Roe scheme)

Parameter	Value	Default
{=}	{= | are | is}	–
epsu	real	0.1

Upwind Method

Syntax

Upwind Method *{=}* *{fourth | gupw | muscl | sucv | upw}* For Equation *EquationString* ([*{of}* *SpeciesName*] | [*{in}* *MaterialPhaseName*])

Summary

Specify method that is blended with pure second order.

Description

This user defined method will be blended with pure second order based on cell Peclet blending. In most cases, this operator is upwinded, however, in the case of 4th order a higher order centered scheme is used.

Parameter	Value	Default
{=}	{= | are | is}	–
UpwMethod	{fourth | gupw | muscl | sucv | upw}	UPW
EquationString	string	–
{of}	{of | species}	–
SpeciesName	string	–
{in}	{in | material_phase}	–
MaterialPhaseName	string	–

Use Approximate Roe Sensitivities

Syntax

Use Approximate Roe Sensitivities

Summary

Use approximate Roe sensitivities for the LHS instead of Van Leer

Description

By default, the Roe flux uses the sensitivities for the Van Leer flux scheme form the LHS matrix. This command turns on a different approximation, based more closely on the Roe flux itself; this approximation converges more quickly for some problems, but is less robust for others. Currently this option only has an effect when the "NC_ADV" equation term is used.

Use Opposing Subface In Open Mass Flux

Syntax

Use Opposing Subface In Open Mass Flux

Summary

Open mass flux will be computed using the interpolated pressure at the opposing subface

Use Specified Pressure In Open Mass Flux

Syntax

Use Specified Pressure In Open Mass Flux

Summary

User specified pressure in open BC will be used to compute gradp in open mass flux BC

Edc Model Specification

Summary

Specify EDC combustion model options.

```
begin Edc Model Specification EdcSpecName

   Activate Co2 Dissociation Model

   Activate Hydrogen Dissociation Model

   Activate Ignition Model {on} IgnPart

   Activate Laminar Limit Model

   Activate Lumped Source Term Model
```

```
   Activate Separate Co Irreversible Oxidation Pathway

   Activation Time {=} Time

   Fuel Name {=} Fuel

   Ignition Threshold Temperature {=} IgnTemp

   Minimum Product Fraction {=} Prmin

   Reaction Time Scale {=} Tchem

   Reference Property {=} Value

   Reference Property Of SpeciesName {=} Value

   begin Oxidizer Mixture Specification OxMixName
   end

end Edc Model Specification EdcSpecName
```

Line Commands

Activate Co2 Dissociation Model

Syntax

Activate Co2 Dissociation Model

Summary

Include effects of CO2 dissociation into CO and O2 at high temperatures

Description

At high temperatures, the equilibrium between CO2, CO, and O2 shifts away from CO2, which can significantly decrease the flame temperature. Activating this model will add this effect to the standard EDC combustion model.

Activate Hydrogen Dissociation Model

Syntax

Activate Hydrogen Dissociation Model

Summary

Include effects of H2 dissociation into H

Description

At temperatures greater than about 2000K, the equilibrium between H2 and H

will yield non-negligible concentrations of H which can significantly decrease flame temperatures. Activating this model will add this effect to the standard EDC combustion model using the correlations of W.W. Erikson, which are derived from the NASA CEA code [62, 63].

Note that the H species must be included in the Cantera input XML file, and neither H nor H2 should be the "last" species in the list since this species is not independent of the rest (to enforce unity sum) and may be susceptible to more noise than the others. Since temperature and other properties are very sensitive to oscillations in the H and H2 equilibrium, this noise could be problematic.

Activate Ignition Model

Syntax

Activate Ignition Model *{on}* *IgnPart*

Summary

MeshPart on which to use the EDC ignition model

Description

Turn on the ignition model for this MeshPart. If fuel and oxidizer are present and the temperature is below the ignition threshold temperature everywhere in the part, reaction will begin. Volume block names and surface names are valid for this command, as well as the alias "all_blocks" and "all_surfaces". Multiple parts must be specified with multiple instances of this line command.

Parameter	Value	Default
{on}	{ @ | at | for | in | on | over}	–
IgnPart	string	–

Activate Laminar Limit Model

Syntax

Activate Laminar Limit Model

Summary

Turn on the EDC laminar limit model for low-turbulence situations

Description

Turn on the EDC laminar limit model. This model requires setting three model constants: CtauLam, CgammaLam, and ClamTrans. The model uses a time scale based on a velocity gradient rather than the turb_ke/turb_diss. This appropriate time scale permits the flame to anchor in laminar regions.

Activate Lumped Source Term Model

Syntax

Activate Lumped Source Term Model

Summary

Nodally lump the EDC source term

Description

If set, the EDC source terms are nodally lumped, rather than using the default implementation of a consistent approach at the Gauss points.

Activate Separate Co Irreversible Oxidation Pathway

Syntax

Activate Separate Co Irreversible Oxidation Pathway

Summary

Add CO oxidation pathway as a separate reaction pathway. This should really be used only in the context of a propellant fire in the presence of hydrogen combustion.

Activation Time

Syntax

Activation Time *{=}* *Time*

Summary

The time at which the EDC combustion model is activated. No combustion will occur before this time.

Parameter	Value	Default
{=}	{= | are | is}	–
Time	real	0.0

Fuel Name

Syntax

Fuel Name *{=}* *Fuel*

Summary

The name of the EDC fuel species, typically either H2 or a major hydrocarbon

Parameter	Value	Default
{=}	{= | are | is}	–
Fuel	string	–

Ignition Threshold Temperature

Syntax

Ignition Threshold Temperature *{=}* *IgnTemp*

Summary

The temperature below which the ignition model is activated

Description

If the ignition model is requested (through the "ACTIVATE IGNITION MODEL" line command, then it will be activated if all temperatures in the corresponding block are below this temperature.

Parameter	Value	Default
{=}	{= | are | is}	–
IgnTemp	real	1000 K

Minimum Product Fraction

Syntax

Minimum Product Fraction *{=} Prmin*

Summary

The minimum product fraction, below which the EDC model will be deactivated.

Parameter	Value	Default
{=}	{= | are | is}	–
Prmin	real	1.0e-6

Reaction Time Scale

Syntax

Reaction Time Scale *{=} Tchem*

Summary

Reaction time scale to set extinction

Description

Characteristic time scale of the chemical kinetics. Residence time in the fine structure region will be compared to this to determine if extinction will result.

Parameter	Value	Default
{=}	{= | are | is}	–
Tchem	real	7.0e-5

Reference

Syntax

Reference *Property {=} Value*

Summary

Reference property for initial diagnostic output

Description

If all required properties are set, the EDC model will print the adiabatic flame

temperature and heat of combustion for the mechanism defined by the specified fuel.

Parameter	Value	Default
Property	string	–
{=}	{= | are | is}	–
Value	real	–

Reference

Syntax

Reference *Property* Of *SpeciesName {=} Value*

Summary

Reference species-dimensioned property for initial diagnostic output

Description

If all required properties are set, the EDC model will print the adiabatic flame temperature and heat of combustion for the mechanism defined by the specified fuel.

Parameter	Value	Default
Property	string	–
SpeciesName	string	–
{=}	{= | are | is}	–
Value	real	–

Oxidizer Mixture Specification

Summary

Specify either mass fractions or mole fractions for the oxidizer mixture. Only non-zero species need to be included, and the mass or mole fractions must sum to unity. The default is air, with a 0.2095:0.7905 molar ratio between O2 and N2.

```
begin Oxidizer Mixture Specification OxMixName

   Mass_Fraction SpeciesName {=} Value

   Mole_Fraction SpeciesName {=} Value

end Oxidizer Mixture Specification OxMixName
```

Line Commands

Mass_Fraction

Syntax

Mass_Fraction *SpeciesName {=} Value*

Summary

Oxidizer mixture mass fraction for the given species

Parameter	Value	Default
SpeciesName	string	–
{=}	{= | are | is}	–
Value	real	–

Mole_Fraction

Syntax

Mole_Fraction *SpeciesName {=} Value*

Summary

Oxidizer mixture mole fraction for the given species

Parameter	Value	Default
SpeciesName	string	–
{=}	{= | are | is}	–
Value	real	–

Hdiff Model Specification

Summary

Specify hydrogen diffusion model options

```
begin Hdiff Model Specification HdiffOptionsName

   Use Reference System Flux {=} {false | no | off | on | true | yes}

end Hdiff Model Specification HdiffOptionsName
```

Line Commands

Use Reference System Flux

Syntax

Use Reference System Flux *{=} {false | no | off | on | true | yes}*

Summary

Use flux defined in the reference (if true) or current (if false) coordinate system.

Description

If the hydrogen flux is written in the current configuration coordinate system, the formulation includes a tensor multiplication by the inverse right Cauchy-Green tensor. This gives the most accurate formulation when the entire equation is being solved in the current configuration. If the flux is written in the reference system (i.e. the parameter value is "true"), no such tensor multiplication is included.

Parameter	Value	Default
{=}	{= | are | is}	–
useReferenceFlux	{false | no | off | on | true | yes}	TRUE

Porous Flow Options

Summary

Specify solution options for porous flow.

```
begin Porous Flow Options blockName

   Integrate Advection By Parts {=} {false | no | off | on | true | yes}

   Use Cvfem {=} {false | no | off | on | true | yes}

   Use Relative Mass Flux {=} {false | no | off | on | true | yes}

end Porous Flow Options blockName
```

Line Commands

Integrate Advection By Parts

Syntax

Integrate Advection By Parts *{=} {false | no | off | on | true | yes}*

Summary

Do you wish to integrate the advection associated with porous flow species by parts?

Parameter	Value	Default
{=}	{= | are | is}	–
integrate_advection_by_parts	{false | no | off | on | true | yes}	–

Use Cvfem

Syntax

Use Cvfem *{=} {false | no | off | on | true | yes}*

Summary

Specify whether to use CVFEM or Galerkin discretization.

Default is Galerkin.

Parameter	Value	Default
{=}	{= | are | is}	–
use_cvfem	{false | no | off | on | true | yes}	–

Use Relative Mass Flux

Syntax

Use Relative Mass Flux *{=} {false | no | off | on | true | yes}*

Summary

Specify whether to use the relative mass flux form of the advection term when mesh motion is on for the porous flow equations. For CVFEM with either the upwind or muscl advection scheme, activating this form will stabilize the mesh motion advection term by using the relative mass flux to determine the upwind direction.

For backwards compatibility, the current default is to NOT use the relative mass flux.

Parameter	Value	Default
{=}	{= | are | is}	–
use_relative_mass_flux	{false | no | off | on | true | yes}	–

Species Options

Summary

Specify solution options for species equations

```
begin Species Options blockName

   Use Cvfem {=} {false | no | off | on | true | yes}

end Species Options blockName
```

Line Commands

Use Cvfem

Syntax

Use Cvfem *{=} {false | no | off | on | true | yes}*

Summary

Specify whether to use CVFEM or Galerkin discretization. Default is Galerkin.

Parameter	Value	Default
{=}	{= | are | is}	–
use_cvfem	{false | no | off | on | true | yes}	–

Turbulence Model Specification

Summary

Specify turbulence modeling options.

```
begin Turbulence Model Specification TurbSpecName

   Activate Cvfem Lumped Turbulent Source Term Algorithm

   Activate Kepsilon Source Term Trickster Linearization

   Activate New Komega Src Linearization

   Activate Rhs Sdr Sensitivities To {dens_sens | sdr_sens | tke_sens |␣
↪turb_prod_sens | tvisc_sens}

   Activate Rhs Tke Sensitivities To {dens_sens | sdr_sens | tke_sens |␣
↪turb_prod_sens | tvisc_sens}
```

```
  Activate Turbulence Clipping Utility At {begin_nonlinear_solve |␣
↪end_nonlinear_solve | manually_run | post_iterate | post_linear_
↪solve | post_nonlinear_solve | pre_iterate | pre_linear_solve | pre_
↪nonlinear_solve | run_initially | run_once | run_post_linear_system_
↪initialization}

  Implicit Les Filter Scale {=} Value

  Limit Turbulent Ke Production To Value Times Dissipation

  Omit Finite Difference Sensitivity Due To Utau

  Omit Sensitivities In Turbulent Production Term

  Omit Sensitivities To Turbulent Production From Ke Source Only

  Omit Velocity Divergence In Turbulent Production Term

  Time Filter {=} Value

  Turbulence Model {=} {dsmag | kepsilon | komega | ksgs | laminar |␣
↪sarans | smag | sst}

  Turbulence Model Parameter {a_1 | beta | beta_1 | beta_star | c_
↪epsilon | c_epsilon_1 | c_epsilon_2 | c_mu | c_mu_cs | c_mu_epsilon␣
↪| edc_c_gamma_lam | edc_c_lam_trans | edc_c_tau_lam | gamma | gamma_1␣
↪| kappa | lr_omega | minimum_turbulent_viscosity | pr_t | ramp_time |␣
↪sa_cb1 | sa_cb2 | sa_cv1 | sa_cw2 | sa_cw3 | sa_sigma | sc_t | sigma_
↪epsilon_lam | sigma_epsilon_trb | sigma_k1_trb | sigma_k_lam | sigma_k_
↪trb | sigma_omega1_trb | sigma_omega_lam | sigma_omega_trb | von_karman␣
↪| wall_length_factor | y_plus_crit} {=} Value

end Turbulence Model Specification TurbSpecName
```

Line Commands

Activate Cvfem Lumped Turbulent Source Term Algorithm

Syntax

Activate Cvfem Lumped Turbulent Source Term Algorithm

Summary

Lump cvfem turbulent ke, tdr and ksgs rhs source terms

Description

This is appropriate for CVFEM only as it uses a special flavor of lumping source terms.

Activate Kepsilon Source Term Trickster Linearization

Syntax

Activate Kepsilon Source Term Trickster Linearization

Summary

Follow classic Patankar approach to linearization of k rhs

Description

The dissipation rate in the k equation includes sensitivities to k by using the Prandtl Kolmorgorov relationship to for the model for of the dissipation rate. This procedure provides additional diagonal dominance for the k rhs source term.

Activate New Komega Src Linearization

Syntax

Activate New Komega Src Linearization

Summary

Alternate komega src linearization

Description

Alt komega src linearization

Activate Rhs Sdr Sensitivities To

Syntax

Activate Rhs Sdr Sensitivities To *{dens_sens | sdr_sens | tke_sens | turb_prod_sens | tvisc_sens}*

Summary

Dial in rhs sens for sdr equation

Parameter	Value	Default
TurbSensParams	{dens_sens | sdr_sens | tke_sens | turb_prod_sens | tvisc_sens}	–

Activate Rhs Tke Sensitivities To

Syntax

Activate Rhs Tke Sensitivities To *{dens_sens | sdr_sens | tke_sens | turb_prod_sens | tvisc_sens}*

Summary

Dial in rhs sens for tke equation

Parameter	Value	Default
TurbSensParams	{dens_sens | sdr_sens | tke_sens | turb_prod_sens | tvisc_sens}	–

Activate Turbulence Clipping Utility At

Syntax

Activate Turbulence Clipping Utility At *{begin_nonlinear_solve | end_nonlinear_solve | manually_run | post_iterate | post_linear_solve | post_nonlinear_solve | pre_iterate | pre_linear_solve | pre_nonlinear_solve | run_initially | run_once | run_post_linear_system_initialization}*

Summary

Activate clipping utility for turbulence quantities

Description

Sometimes it helps to clip the turbulence quantities. This is a utility that provides this support.

Parameter	Value	Default
UtilClipLocation	{begin_nonlinear_solve | end_nonlinear_solve | manually_run | post_iterate | post_linear_solve | post_nonlinear_solve | pre_iterate | pre_linear_solve | pre_nonlinear_solve | run_initially | run_once | run_post_linear_system_initialization}	–

Implicit Les Filter Scale

Syntax

Implicit Les Filter Scale *{=}* *Value*

Summary

Provide implicit filter length

Description

In constant coefficient LES models, allow the user to specify and implicit filter length. The default is to determine a length scale tied to the mesh.

Parameter	Value	Default
{=}	{= | are | is}	–
Value	real	–

Limit Turbulent Ke Production

Syntax

Limit Turbulent Ke Production To *Value* Times Dissipation

Summary

Choose to limit production source terms

Description

This option limits the turbulent ke production to a scale factor of dissipation, prod = min(prod, limit*den*en1). In practice, the ratio of production to dissipation is not very high. In some flows, it is useful to specify a value of approximately 1000. The ratio should be checked as part of the analysis to make sure that violation of the physical ratio has not been done. In general, this option is only activated in domain locations where dissipation rate is very small.

Parameter	Value	Default
Value	real	1.0e-8

Omit Finite Difference Sensitivity Due To Utau

Syntax

Omit Finite Difference Sensitivity Due To Utau

Summary

Omit FD sensitivity from utau

Description

The wall friction velocity, utau, can be complex to evaluate in the context of the law of the wall. The default is to compute sensitivities via finite difference. Sometimes, this seems to add some overhead and spurious sensitivities.

Omit Sensitivities In Turbulent Production Term

Syntax

Omit Sensitivities In Turbulent Production Term

Summary

Do not include sensitivities for turbulence production

Description

This option removes the analytical sensitivities for the production of turbulent kinetic energy. It seems that most simulations are more stable if this option is activated.

Omit Sensitivities To Turbulent Production From Ke Source Only

Syntax

Omit Sensitivities To Turbulent Production From Ke Source Only

Summary

Omit sensitivities to turbulence production from the KE source term

Description

This option removes the analytical sensitivities for the production of turbulent kinetic energy in the KE source terms (where they may be destabilizing) but retains them elsewhere (where they may be helpful). Mutually exclusive w/ the previous option.

Omit Velocity Divergence In Turbulent Production Term

Syntax

Omit Velocity Divergence In Turbulent Production Term

Summary

Do not include divergence term in turbulence production

Description

This option removes the divergence term from the turbulent production of kinetic energy. The default is to include this term.

Time Filter

Syntax

Time Filter *{=}* *Value*

Summary

Time filter size for Time Filtered Navier-Stokes model

Description

Turbulent viscosity is normally calculated based on a time scale given by k/epsilon (for k-epsilon models) or T (v2f model). The TFNS model substitutes the minimum of the normal computed value and the user-specified time filter size in the turbulent viscosity calculation. In general, this filter should be no less than twice the physical time step. A non-fatal warning is issued if this condition is violated.

Parameter	Value	Default
{=}	{= | are | is}	–
Value	real	1.0e32

Turbulence Model

Syntax

Turbulence Model *{=}* *{dsmag* | *kepsilon* | *komega* | *ksgs* | *laminar* | *sarans* | *smag* | *sst}*

Summary

Specify type of turbulence model to be used

Parameter	Value	Default
{=}	{= | are | is}	–
TurbulenceModel	{dsmag | kepsilon | komega | ksgs | laminar | sarans | smag | sst}	–

Turbulence Model Parameter

Syntax

Turbulence Model Parameter *{a_1* | *beta* | *beta_1* | *beta_star* | *c_epsilon* |

c_epsilon_1 | *c_epsilon_2* | *c_mu* | *c_mu_cs* | *c_mu_epsilon* | *edc_c_gamma_lam* | *edc_c_lam_trans* | *edc_c_tau_lam* | *gamma* | *gamma_1* | *kappa* | *lr_omega* | *minimum_turbulent_viscosity* | *pr_t* | *ramp_time* | *sa_cb1* | *sa_cb2* | *sa_cv1* | *sa_cw2* | *sa_cw3* | *sa_sigma* | *sc_t* | *sigma_epsilon_lam* | *sigma_epsilon_trb* | *sigma_k1_trb* | *sigma_k_lam* | *sigma_k_trb* | *sigma_omega1_trb* | *sigma_omega_lam* | *sigma_omega_trb* | *von_karman* | *wall_length_factor* | *y_plus_crit} {=} Value*

Summary

Turbulence model parameters

Parameter	Value	Default
TurbParams	{a_1 | beta | beta_1 | beta_star | c_epsilon | c_epsilon_1 | c_epsilon_2 | c_mu | c_mu_cs | c_mu_epsilon | edc_c_gamma_lam | edc_c_lam_trans | edc_c_tau_lam | gamma | gamma_1 | kappa | lr_omega | minimum_turbulent_viscosity | pr_t | ramp_time | sa_cb1 | sa_cb2 | sa_cv1 | sa_cw2 | sa_cw3 | sa_sigma | sc_t | sigma_epsilon_lam | sigma_epsilon_trb | sigma_k1_trb | sigma_k_lam | sigma_k_trb | sigma_omega1_trb | sigma_omega_lam | sigma_omega_trb | von_karman | wall_length_factor | y_plus_crit}	–
{=}	{= | are | is}	–
Value	real	–

7.41 Toggle

Toggle Block

Scope

Sierra

Summary

This command block defines the period(s), state and initial conditions for model feature toggling.

Description

Defines a generic entity toggling specification and associate it with name *ModelName*. The same *ModelName* can be used for various model features such as element blocks, boundary conditions, contact and volumetric sources.

```
begin Toggle Block ModelName

   Freeze Element Block Solution State
```

```
   Period {=} Period_list...

   State {=} {active | inactive}

end Toggle Block ModelName
```

Line Commands

Freeze Element Block Solution State

Syntax

Freeze Element Block Solution State

Summary

Specify that the solution state remain the same when an element block has been toggled on.

Description

Specifies that the initial condition is not reset on any associated element block that has been toggled. The default behavior is to have the initial conditions reset on element blocks that have been toggled back on.

Period

Syntax

Period *{=} Period_list. . .*

Summary

Specify the periods over which this toggling model is to be used.

Description

Defines the time periods over which this toggling model is to be used. Note that more than one PERIOD can be appear in the command block.

Parameter	Value	Default
{=}	{= | are | is}	–
Period_list	string. . .	–

State

Syntax

State *{=} {active | inactive}*

Summary

Specify the state for this toggling model over the periods it is to be used.

Description

Defines the toggle state for the time periods over which this toggling model is to be used. Note that more than one STATE can be appear in the command block.

Parameter	Value	Default
{=}	{= | are | is}	–
ToggleState	{active | inactive}	–

Use Toggle Block

Syntax

Use Toggle Block *ToggleName* [*{on} ElementBlockList...*]

Summary

Specification for toggling entities in the computational model based on Toggle Block parameters. When used at the region level, the list of element blocks to be toggled must be provided. Otherwise a listing of entities is not needed as the Toggle Block will be associated with the command line or the enclosing command block.

Parameter	Value	Default
ToggleName	string	–

7.42 Submodel

Submodel

Scope

Aria Region, Equation System, Explicit Equation System, Root Finder Equation System

Summary

Contains the commands needed to define an Aria submodel.

Description

This block command is used to define the embedded block submodel.

```
begin Submodel SubModel_name

   Clip Isoparametric Coords

   Embedded Blocks {=} VolumeList...
```

```
   Enclosing Blocks {=} VolumeList...

   Expand Box Percentage {=} Percentage

   Field Name {=} FieldName

   Interaction Output {=} {active | all | error | none | summary}

   Search Method {=} {boost | kdtree}

   Search Tolerance {=} Tolerance

end Submodel SubModel_name
```

Line Commands

Clip Isoparametric Coords

Syntax

Clip Isoparametric Coords

Summary

This command is no longer used. It remains available for backwards compatibility until version 4.50.

Embedded Blocks

Syntax

Embedded Blocks *{=} VolumeList...*

Summary

Specifies the element blocks to be embedded within another volume.

Parameter	Value	Default
{=}	{= | are | is}	–
VolumeList	string...	–

Enclosing Blocks

Syntax

Enclosing Blocks *{=} VolumeList...*

Summary

Specifies the element blocks in which EMBEDDED BLOCKS can reside.

Parameter	Value	Default
{=}	{= | are | is}	–
VolumeList	string. . .	–

Expand Box Percentage

Syntax

Expand Box Percentage *{=}* *Percentage*

Summary

This command is no longer used. It remains available for backwards compatibility until version 4.50.

Parameter	Value	Default
{=}	{= | are | is}	–
Percentage	real	1.0

Field Name

Syntax

Field Name *{=}* *FieldName*

Summary

Specifies the fields to tie constraints to (default is temperature).

Parameter	Value	Default
{=}	{= | are | is}	–
FieldName	string	–

Interaction Output

Syntax

Interaction Output *{=}* *{active | all | error | none | summary}*

Summary

When this line is present, detailed information of the contact interactions are output.

Parameter	Value	Default
{=}	{= | are | is}	–
NonconformalOutput	{active | all | error | none | summary}	NONE

Search Method

Syntax

Search Method *{=}* *{boost | kdtree}*

Summary

Set the search method to be used with nonconformal contact.

Parameter	Value	Default
{=}	{= | are | is}	–
SearchMethod	{boost | kdtree}	NONE

Search Tolerance

Syntax

Search Tolerance *{=}* *Tolerance*

Summary

Assign the normal search tolerance in the length units of the model. This will override the automatic tolerancing.

Parameter	Value	Default
{=}	{= | are | is}	–
Tolerance	real	1.0

8 Symbols

Table 8.1: List Of Symbols

Symbol	Description	First Appearance
F_{ext}	Extension speed	*Level Set*
H_ε	Smooth Heaviside	*Level Set*
R	Solution Residual	*Governing Equations*
$\tilde{\mathcal{M}}(\mathcal{A})^T \tilde{\mathcal{M}}(\mathcal{A})$	Useful energy norm measure for convergence result	*Teko Heuristic Multiphysics Pre*
Ω	Volumetric Domain	*Domain Definition*
Φ	Level set variable	*Level Set*
$\mathbf{x}$	Spatial Coordinate Vector	*Domain Definition*
$\frac{\partial H_\varepsilon}{\partial \Phi}$	Smooth Heaviside	*Level Set*
κ	Level set curvature	*Level Set*
$\mathcal{RAC}$	Blocked matrix with simultaneous row/column reordering	*Linear Ordering Problem for B*
$\mathcal{A}$	Blocked matrix	*Teko Heuristic Multiphysics Pre*
$\mathcal{M}(\mathcal{A})$	Blocked matrix	*Choosing Sub-block Solvers for*
S	Generic Surface	*Domain Definition*
V	Generic Volume	*Domain Definition*
$\partial\Omega$	Boundary of domain Ω	*Domain Definition*
$\partial\Omega_{i-j}$	Interface of domains Ω_i and Ω_j	*Domain Definition*
ϕ	Solution Variables	*Governing Equations*
ψ	Finite Element Weight Function	*Governing Equations*
σ	Surface tension coefficient	*Level Set/CDFEM*
$\boldsymbol{J}$	Jacobian Matrix	*Solution Strategy*
$\tilde{\mathcal{M}}(\mathcal{RAC})$	Blocked matrix with simultaneous row/column reordering	*Linear Ordering Problem for B*
$\tilde{\mathcal{M}}(\mathcal{A})$	Blocked matrix	*Teko Heuristic Multiphysics Pre*
$\tilde{\mathcal{M}}(\mathcal{A}')$	Blocked matrix with simultaneous row/column reordering	*Block Merging Problem for Blo*
ε	Smooth Heaviside interface thickness	*Level Set*
$\boldsymbol{F_\sigma}$	Surface tension force vector	*Level Set/CDFEM*
$\boldsymbol{I}$	Identity tensor	*Level Set/CDFEM*
$\boldsymbol{T_\sigma^{stab}}$	Surface tension stress tensor stabilization	*Level Set/CDFEM*
$\boldsymbol{T_\sigma}$	Surface tension stress tensor	*Level Set/CDFEM*
$\boldsymbol{\Delta\phi}$	Solution Increment	*Solution Strategy*
$\boldsymbol{R}$	Residual Vector	*Solution Strategy*
$\boldsymbol{\phi}$	Solution Vector	*Solution Strategy*
$\boldsymbol{f_\sigma}$	Surface tension force vector	*Level Set/CDFEM*
$\boldsymbol{n}$	Level set normal	*Level Set*

Table 8.2: General Notation

Expression	Meaning
$\mathring{\mathcal{R}}$	interior of $\mathcal{R}$
$\partial\mathcal{R}$	boundary of $\mathcal{R}$
$\mathcal{R} \cup \mathcal{F}$	union of $\mathcal{R}$ and $\mathcal{F}$
$\mathcal{R} \cap \mathcal{F}$	intersection of $\mathcal{R}$ and $\mathcal{F}$
$\mathcal{R} \subset \mathcal{F}$	$\mathcal{R}$ is a subset of $\mathcal{F}$
$\mathcal{R} \setminus \mathcal{F}$	set complement
$x \in \mathcal{R}$	x is an element of the set $\mathcal{R}$
$f : \mathcal{R} \to \mathcal{F}$	f maps the set $\mathcal{R}$ into the set $\mathcal{F}$; $\mathcal{R}$ is the domain, $\mathcal{F}$ the codomain
$x \mapsto f(x)$	mapping that carries x into $f(x)$; e.g. $x \mapsto x^2$ is the mapping that carries every real number x into its square
$f \circ g$	composition of the mappings f and g; i.e. $(f \circ g)(x) = f(g(x))$
$\{x \mid R(x) \text{ holds}\}$	the set of all x such that $R(x)$ holds; e.g. $\{x \mid 0 \leq x \leq 1\}$ is the interval $[0, 1]$
$\delta(\mathbf{x} - \mathbf{x}_0)$	Dirac delta function centered at $\mathbf{x}_0$
$[\![\cdot]\!]$	measure of jump in a quantity across an interface
$\mathbb{R}$	real numbers
$\mathbb{N}$	natural numbers

This page intentionally left blank.

References

[1] Patrick K Notz, Roger P Pawlowski, and James C Sutherland. Graph-based software design for managing complexity and enabling concurrency in multiphysics pde software. *ACM Transactions on Mathematical Software (TOMS)*, 39(1):1–21, 2012.

[2] Michael L. Hobbs, Judith A. Brown, Michael J. Kaneshige, and Cuauhtemoc Aviles-Ramos. Cookoff of powdered and pressed explosives using a micromechanics pressurization model. *Propellants, Explosives, Pyrotechnics*, 47(2):e202100156, 2022. URL: https://onlinelibrary.wiley.com/doi/abs/10.1002/prep.202100156, arXiv:https://onlinelibrary.wiley.com/doi/pdf/10.1002/prep.202100156, doi:https://doi.org/10.1002/prep.202100156.

[3] W. M. Deen. *Analysis of Transport Phenomena*. Topics in Chemical Engineering. Oxford University Press, New York, 1998.

[4] Lawrence E. Malvern. *Introduction to the Mechanics of a Continuous Medium*. Series in Engineering of the Physical Sciences. Prentice-Hall, Upper Saddle River, NJ, USA, 1969.

[5] George E. Mase. *Theory and Problems of Continuum Mechanics*. Schaum's Outline Series. McGraw-Hill, New York, NY, USA, 1970.

[6] Javier Bonet and Richard D. Wood. *Nonlinear Continuum Mechanics for Finite Element Analysis*. Cambridge University Press, 1997.

[7] Ted Belytschko, Wing Kam Liu, and Brian Moran. *Nonlinear Finite Elements for Continua and Structures*. John Wiley and Sons, 2004.

[8] M. J. Martinez. Mathematical and numerical formulation of nonisothermal multicomponent three-phase flow in porous media. Technical Report SAND95-1247, Sandia National Laboratories, Albuquerque, NM, USA, 1995. http://www.prod.sandia.gov/cgi-bin/techlib/access-control.pl/1995/951247.pdf.

[9] M. J. Martinez, P. L. Hopkins, and J. N. Shadid. LDRD final report: physical simulation of nonisothermal multiphase multicomponent flow in porous media. Technical Report SAND97-1766, Sandia National Laboratories, Albuquerque, NM, USA, 1997. http://www.prod.sandia.gov/cgi-bin/techlib/access-control.pl/1997/971766.pdf.

[10] M. J. Martinez, P. L. Hopkins, and P. N. Reeves. PorSalsa user's manual. Technical Report SAND2001-1555, Sandia National Laboratories, Albuquerque, NM, USA, 2001. http://www.prod.sandia.gov/cgi-bin/techlib/access-control.pl/2001/011555.pdf.

[11] Chris Lautenberger and Carlos Fernandez-Pello. Generalized pyrolysis model for combustible solids. *Fire Safety Journal*, 44(6):819–839, 2009.

[12] D. K. Gartling, C. E. Hickox, and R. C. Givler. Simulations of coupled viscous and porous flow problems. *Comp. Fluid Dyn.*, 1-2:23–48, 1997.

[13] S. A. Roberts, D. R. Noble, E. M. Benner, and P. R. Schunk. Multiphase hydrodynamic

lubrication flow using a three-dimensional shell finite element model. *Comput. Fluids*, 2013. , in press.

[14] Ely M Gelbard. Application of spherical harmonics method to reactor problems. *Bettis Atomic Power Laboratory, West Mifflin, PA, Technical Report No. WAPD-BT-20*, 1960.

[15] A. D. Kraus and A. Bar-Cohen. *Thermal Analysis and Control of Electronic Equipment*. Hemisphere, Washington DC, 1983.

[16] J. H. Lienhard IV and J. H. Lienhard V. *A Heat Transfer Textbook*. Phlogiston Press, Cambridge, Massachusetts, 3rd edition, 2004. URL: http://web.mit.edu/lienhard/www/ahtt.html.

[17] C. V. Madhusudana. *Thermal Contact Conductance*. Springer-Verlag, New York, 1996.

[18] R. Siegel and J. R. Howell. *Thermal Radiation Heat Transfer*. Taylor and Francis, Washington, D.C., 1992.

[19] M. W. Glass. Chaparral: a library for solving enclosure radiation heat transfer problems. Technical Report SAND01-xxxx, Sandia National Laboratories, Albuquerque, New Mexico, 2001.

[20] B. R. Carnes and K. D. Copps. Thermal contact algorithms in sierra mechanics. Technical Report SAND2008-2607, Sandia National Laboratories, Albuquerque, New Mexico, April 2008.

[21] D.N. Arnold, B. Cockburn, and L. D. Marini. Unified analysis of discontinuous galerkin methods for elliptic problems. *SIAM J. Numer. Anal.*, 39:1749–79, 2002.

[22] E. B. Becker, G. F. Carey, and J. T. Oden. *Finite Elements: An Introduction*. Prentice-Hall Inc., New Jersey, 1981.

[23] T. R. Young. Chemeq—a subroutine for solving stiff ordinary differential equations. Technical Report NRL Memorandum Report 4091, Naval Research Laboratory, Washington D.C., 1980.

[24] B. A. Finlayson. *The Method of Weighted Residuals and Variational Principles*. Academic Press, New York, 1972.

[25] P. Gresho, R. Lee, R. Sani, and T. Stullich. On the time-dependent FEM solution of the incompressible navier-stokes equations in two and three dimensions. *Recent Advances in Numerical Methods in Fluids*, 1980.

[26] D. K. Gartling, R. E. Hogan, and M. W. Glass. Coyote—a finite element computer program for nonlinear heat conduction problems, part I—theoretical background, version 4.05. Technical Report SAND94-1173, Sandia National Laboratories, Albuquerque, New Mexico, 2000.

[27] D. K. Gartling, R. E. Hogan, and M. W. Glass. Coyote—a finite element computer program for nonlinear heat conduction problems, part II—user's manual, version 4.05. Technical Report SAND94-1179, Sandia National Laboratories, Albuquerque, New Mexico, 2000.

[28] J. E. Marsden and A. J. Tromba. *Vector Calculus*. W. H. Freeman and Company, New York, 1981.

[29] S. Domino. Design-order, non-conformal low-Mach fluid algorithms using a hybrid CVFEM/DG approach. *J. Comp. Phys*, 359:331–351, 2018.

[30] S. Domino, P. Sakievich, and M. Barone. A assessment of atypical mesh topologies for low-Mach large-eddy simulation. *Comp. Fluids*, 179:655–669, 2019.

[31] Robert Moser, editor. *Numerical Methods in Turbulence Simulation*. Computation and Analysis of Turbulent Flows. Academic Press, San Diego, CA, December 2022.

[32] Guglielmo Scovazzi and T Hughes. Lecture notes on continuum mechanics on arbitrary moving domains. *Lecture Notes, November*, 2007.

[33] Eric C Cyr, John N Shadid, and Raymond S Tuminaro. Teko: a block preconditioning capability with concrete example applications in navier–stokes and mhd. *SIAM Journal on Scientific Computing*, 38(5):S307–S331, 2016.

[34] Rafael Martí and Gerhard Reinelt. *The linear ordering problem: exact and heuristic methods in combinatorial optimization*. Volume 175. Springer Science & Business Media, 2011.

[35] Irene Charon and Olivier Hudry. A branch-and-bound algorithm to solve the linear ordering problem for weighted tournaments. *Discrete Applied Mathematics*, 154(15):2097–2116, 2006.

[36] G. S. Beavers and D. D. Joseph. Boundary conditions at a naturally permeable wall. *J. Fluid Mech.*, 30:197–207, 1967.

[37] P. G. Saffman. On the boundary condition at the surface of a porous medium. *Studies in Applied Mathematics*, 50(2):93–101, 1971.

[38] R. H. Davis and H. A. Stone. Flow through beds of porous particles. *Chemical Engineering Science*, 48(23):3993–4005, 1993.

[39] James A Sethian. A fast marching level set method for monotonically advancing fronts. *Proceedings of the National Academy of Sciences*, 93(4):1591–1595, 1996.

[40] David R Noble, Elijah P Newren, and Jeremy B Lechman. A conformal decomposition finite element method for modeling stationary fluid interface problems. *International Journal for Numerical Methods in Fluids*, 63(6):725–742, 2010.

[41] Bruno Lafaurie, Carlo Nardone, Ruben Scardovelli, Stéphane Zaleski, and Gianluigi Zanetti. Modelling merging and fragmentation in multiphase flows with surfer. *Journal of computational physics*, 113(1):134–147, 1994.

[42] Richard A Cairncross, P Randall Schunk, Thomas A Baer, Rekha R Rao, and Phillip A Sackinger. A finite element method for free surface flows of incompressible fluids in three dimensions. part i. boundary fitted mesh motion. *International journal for numerical methods in fluids*, 33(3):375–403, 2000.

[43] S Hysing. A new implicit surface tension implementation for interfacial flows. *International Journal for Numerical Methods in Fluids*, 51(6):659–672, 2006.

[44] D. Dobranich. Safsim input manual -a compute program for the engineering simulation of flow systems. Technical Report SAND92-0694, Sandia National Laboratories, Albuquerque, NM, USA, September 1992.

[45] Eric C Cyr, John N Shadid, and Raymond S Tuminaro. Stabilization and scalable block preconditioning for the navier–stokes equations. *Journal of Computational Physics*, 231(2):345–363, 2012.

[46] Youcef Saad. A flexible inner-outer preconditioned gmres algorithm. *SIAM Journal on Scientific Computing*, 14(2):461–469, 1993.

[47] Robert Kosik, Peter Fleischmann, Bernhard Haindl, Paola Pietra, and Siegfried Selberherr. On the interplay between meshing and discretization in three-dimensional diffusion simulation. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 19(11):1233–1240, 2000.

[48] Changna Lu, Weizhang Huang, and Jianxian Qiu. Maximum principle in linear finite element approximations of anisotropic diffusion–convection–reaction problems. *Numerische Mathematik*, 127(3):515–537, 2014.

[49] Dmitri Kuzmin. Linearity-preserving flux correction and convergence acceleration for constrained galerkin schemes. *Journal of Computational and Applied Mathematics*, 236(9):2317–2337, 2012.

[50] Dmitri Kuzmina and John N Shadidb. A new approach to enforcing discrete maximum principles in continuous galerkin methods for convection-dominated transport equations. *Preprint: Ergebnisberichte des Instituts für Angewandte Mathematik*, 2015.

[51] I. M. Levi. Users' Guide to the Transient Heat Conduction Finite Element Code HTCON. Technical Report MM70-5425-17, Bell Telephone Laboratories, Whippany, NJ, USA, 1970.

[52] A. F. Emery, K. Sugihara, and A. T. Jones. A comparison of some of the thermal characteristics of finite-element and finite-difference calculations of transient problems. *Num. Heat Trans.*, 2:97–113, 1979.

[53] F. Damjanic and D. R. J. Owen. Practical considerations for thermal transient finite element analysis using isoparametric elements. *Nucl. Engrg. Des.*, 69:109–126, 1982.

[54] E. Rank, C. Katz, and H. Werner. On the importance of the discrete maximum principle in transient analysis using finite element methods. *Int. J. Num. Meth. Engng.*, 19:1771–1782, 1983.

[55] H. S. Carslaw and J. C. Jaeger. *Conduction of Heat in Solids*. Clarendon Press, Oxford, 2\textsuperscript nd edition, 1959.

[56] I. M. Krieger. Rheology of monodisperse latices. *Adv. Colloid Interface Sci.*, 3:111–136, 1972.

[57] D. K. Gartling. NACHOS II - a finite element computer program for incompressible flow problems. part I - theoretical background. Technical Report SAND86-1816, Sandia National Labs, Albuquerque, NM, USA, April 1986.

[58] E. R. G. Eckert. Survey of boundary layer heat transfer at high velocities and high temperatures. Technical Report, WADC, 1960.

[59] Clark R. Dohrmann and Pavel B. Bochev. A stabilized finite element method for the Stokes problem based on polynomial pressure projections. *Int. J. Num. Meth. Fluids*, 46:183–201, 2004.

[60] Pavel B. Bochev, Clark R. Rohrmann, and Max D. Gunzburger. Stabilization of low-order mixed finite elements for the Stokes equations. *SIAM J. Numer. Anal.*, 44(1):82–101, 2006.

[61] T. Hughes, L. P. Franca, and M. Balestra. A new finite element formulation for computational fluid dynamics: v. circumventing the babuska-brezzi condition: a stable petrov-galerkin formulation of the stokes problem accomodating equal-order interpolations. *Comput. Methods Appl. Mech. Engrg.*, 59:85–99, 1986.

[62] S. Gordan and B. J. McBride. Computer program for calculation of complex chemical equilibrium compositions and applications I. analysis. Technical Report Ref. Publication 1311, NASA, October 1994.

[63] B. J. McBride and S. Gordan. Computer program for calculation of complex chemical equilibrium compositions and applications II. users manual and program description. Technical Report Ref. Publication 1311, NASA, June 1996.

This page intentionally left blank.

Index

C

Cauchy stress, 66
contact
    resistance, 71
    tied, 71

D

deformation gradient, 65

E

Equation
    Ampere, 43
    Brinkman Momentum, 56
    Charge Density, 43
    Continuity, 36
    Current, 42
    Energy, 37
    Fluid Momentum, 39
    Level Set, 63
    Lubrication, 57
    Mass Conservation, 36
    Potential Projection Equation, 59
    Radiation Transport, 41
    Solid Momentum, 40
    Species, 38
    Stress Tensor Projection, 58
    Suspension, 44
    Voltage, 42

G

Gaussian quadrature, 97
Goma, 10
Green strain, 65
Green-Lagrange strain, 65

J

Jacobian matrix, 96

L

Lagrangian strain, 65
Lamé coefficients, 66

M

master element, 96

P

Piola-Kirchhoff stress, 66
Poisson's ratio, 66
porosity, 45
PSPG, 6493
PSPP, 6493

R

reference element, 96
residual stress, 66

S

shear modulus, 66
stabilization, 6493
stiffness matrix, 95
SUPG, 38

T

thermal stress, 66

U

unit conversions, 15
units, 15

V

void ratio, 45
volumetric source, 70

Y

Young's modulus, 66

Sandia National Laboratories is a multimission laboratory managed and operated by National Technology & Engineering Solutions of Sandia LLC, a wholly owned subsidiary of Honeywell International Inc., for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-NA0003525.