SAND2026-20879

SANDIA REPORT

Printed May 4, 2026

Sandia National Laboratories

Sierra/SD – Its2Sierra – User's Manual – 5.30

Sierra Structural Dynamics Development Team

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

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

Available to DOE and DOE contractors from

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

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

Available to the public from

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

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

ABSTRACT

The Integrated Tiger Series (ITS) generates a database containing energy deposition data. This data, when stored on an **Exodus** file, is not typically suitable for analysis within **Sierra Mechanics** for finite element analysis. The **its2sierra** tool maps data from the ITS database to the **Sierra** database.

This document provides information on the usage of **its2sierra**.

This page left blank

CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACKNOWLEDGMENTS

The **Sierra/SD** software package is the collective effort of many individuals and teams. A core Sandia National Laboratories based **Sierra/SD** development team is responsible for maintenance of documentation, testing, and support of code capabilities. This team includes Dagny Beale, Gregory Bunting, Nate Crane, David Day, Clark Dohrmann, Payton Lindsay, Justin Pepe, Julia Plews, Jesse Thomas, Ben Treweek, and Johnathan Vo.

The **Sierra/SD** team also works closely with the Sierra FuSED and Plato teams to jointly enhance and maintain several capabilities. This includes contributions from Wilkins Aquino, Sean Hardesty, Elizabeth Livingston, Chandler Smith, Timothy Walsh, and Ray Wildman.

The **Sierra/SD** team works closely with other Sierra teams on core libraries and shared tools. This includes the DevOps, Sierra Toolkit, Solid Mechanics, Fluid Thermal Teams. Additionally, analysts regularly provide code capabilities as well as help review and verify code capabilities, testing, and documentation. Other individuals not already mentioned directly contributing to the **Sierra/SD** documentation, testing, and code base during the last year include Victor Brunini, Jonathan Clausen, David Glaze, Mark Hamilton, Andrew Kimler, Dong Lee, Kevin Manktelow, Mark Merewether, Scott Miller, Matthew Mosby, Tony Nguyen, Tolu Okusanya, Heather Pacella, Kendall Pierson, Nick Reynolds, Timothy Shelton, John Shimanek, Greg Sjaardema, Clinton Stimpson, Tyler Voskuilen, Ellen Wagman, Alan Williams, and Christopher Wilson.

Historically dozens of other Sandia staff, students, and external collaborators have also contributed to the **Sierra/SD** product and its documentation.

Many other individuals groups have contributed either directly or indirectly to the success of the **Sierra/SD** product. These include but are not limited to;

- Garth Reese implemented the original **Sierra/SD** code base. He served as principal investigator and product owner for **Sierra/SD** for over twenty years. His efforts and contributions led to much of the current success of **Sierra/SD**.
- Clark Dohrmann who developed the GDSW solver that is vital for parallel and GPU performance of **Sierra/SD**.
- David Day who directly supported **Sierra/SD** development work for over twenty years.
- The ASC program at the DOE which funded the initial **Sierra/SD** (Salinas) development as well as the ASC program which still provides the bulk of ongoing development support.
- Line managers at Sandia Labs who supported this effort. Special recognition is extended to David Martinez who helped establish the effort.

- Charbel Farhat and the University of Colorado at Boulder. They have provided incredible support in the area of finite elements, and especially in development of linear solvers.
- Carlos Felippa of U. Colorado at Boulder. His consultation has been invaluable, and includes the summer of 2001 where he visited at Sandia and developed the Hexshell element for us.
- Danny Sorensen, Rich Lehoucq and other developers of ARPACK, which is used for eigenvalue problems.
- Esmond Ng who wrote *sparspak* for us. This sparse solver package is responsible for much of the performance in **Sierra/SD** linear solvers.
- The *metis* team at the University of Minnesota. *Metis* is an important part of the graph partitioning schemes used by several of our linear solvers. These are copyright 1997 from the University of Minnesota.
- Padma Raghaven for development of a parallel direct solver that is a part of the linear solvers.
- The developers of the SuperLU Dist parallel sparse direct linear solver. It is used through GDSW for a variety of problems.
- Leszek Demkowicz at the University of Texas at Austin who provided the HP3D[2] library and has worked with the **Sierra/SD** team on several initiatives. The HP3D library is used to calculate shape functions for higher order elements.

  This work was supported by the Laboratory Directed Research and Development (LDRD) program.

1. INTRODUCTION

The Integrated TIGER Series (ITS)[1] generates a database containing energy deposition data. This data, when stored on an **Exodus** file, is not typically suitable for including in a SIERRA finite element analysis without additional processing.

The ITS output database contains a set of ITS zones (represented by **Exodus** blocks) and ITS subzones (represented as **Exodus** elements), each with an energy deposition (`EDEP`) and volume (`VOL`) value assigned. Although **Exodus** is the input database standard for SIERRA analysis, the ITS output **Exodus** files usually have a different mesh description from the one that is typically used for SIERRA runs.

The **Its2Sierra** tool can be used to map data from the ITS **Exodus** database subzones to the SIERRA database elements in order to make the handoff analysis more robust and accurate.

The **Its2Sierra** product is under active development by the **Sierra/SD** team. If additional features are needed, they can be requested by submitting an email to sierra-help@sandia.gov.

This page intentionally left blank.

2. RELEASE NOTES

The sections in this chapter mainly describe new features, bug fixes, performance improvements, and features deprecated or removed in each new version of **Its2Sierra**.

2.1. Release 5.22

New or Improved Features

- The **Its2Sierra** tool now supports interpolation to the nearest Gauss point on the output SIERRA mesh, based upon an inverse-distance weighting of values from the source ITS mesh. Refer to section 5.2.3 for additional details on the interpolation algorithm and table 4-1 for an explanation of how to use it.
- **Its2Sierra** now preserves variables, attributes, names, node numbering, QA information, and more, on the input SIERRA mesh, as detailed in section 4.2.
- An option to conserve energy between ITS and SIERRA meshes is now available. Consult section 5.3 for details.
- Significant additional testing and verification has been added to **Its2Sierra** over the past release cycle.

Bug Fixes

- A bug has been addressed in tetrahedral volume calculation which caused lower energy values than expected on tetrahedral meshes.

This page intentionally left blank.

3. RUNNING ITS2SIERRA

Its2Sierra is a command line tool which requires the user to supply a valid **Exodus** file containing 'EDEP' and 'VOL' element variables, an **Exodus** file representing the SIERRA mesh, and a special input file for **Its2Sierra** itself. A new **Exodus** database containing the SIERRA mesh with mapped ITS 'EDEP' and the SIERRA 'Volume' is created after successful execution.

Its2Sierra currently only supports serial execution on a single processor unless using the '--beta' option. See Chapter 6 for additional information.

The **Its2Sierra** tool is invoked on the command line by

```
$ its2sierra <its_output.exo> <sierra_mesh.exo> <output.exo> <Its2Sierra.inp>
```

where

its_output.exo is the output **Exodus** file from an ITS run. It must contain EDEP and VOL data on each element.

sierra_mesh.exo is the target **Exodus** mesh to be used for analysis in **Sierra/SD** or **Sierra/SM**.

output.exo is the output **Exodus** mesh with the same mesh/geometry as sierra_mesh.exo which also includes the EDEP data from **its_output.exo** and calculated element volumes. Many more details on the output format are available in Section 4.2.

Its2Sierra.inp is a special **Its2Sierra** input file, described in detail in Section 4.1.

The names of each of these input and output files may be specified on the command line, but for simplicity, they are referred to by these sample names for the remainder of the document.

This page intentionally left blank.

4. INPUT AND OUTPUT

This chapter details the structure of the **Its2Sierra** input file and the expected output for use in a SIERRA analysis.

4.1. Input file syntax

The **Its2Sierra** input file is used to specify mapping (sets of) blocks of elements between the ITS mesh and the SIERRA mesh. Each section of the input file specifies an ITS zone or block (or set of zones/blocks) for which energies are then mapped to a SIERRA element block (or set of blocks). Table 4-1 lists the options that can be used to control the mapping parameters in each block, as well as their defaults, where applicable. Each section of the file begins with the keyword 'ITSBlocks' and ends with the keyword 'END'. Multiple sections may exist in the input file, and the text parsing is case-insensitive.

Parameter	Type	Default	Description
ITSBlocks	block list		Begins a section of **Its2Sierra** input and specifies blocks in the ITS input file to be included in this section.
Mapping	blockwise| nearest_neighbor| interpolation	blockwise	Type of mapping to be used in this set. Consult section 5.2 for details on each algorithm.
Sierra_Blocks	block list		Lists SIERRA blocks to be used in this set.
Energy_Scale_Factor	scalar	1.0	User-specified scalar factor which is applied uniformly to mapped energies.
Mesh_Scale_Factor	scalar	1.0	User-specified scalar factor which is applied uniformly to ITS mesh coordinates.
Conserve_Total_Energy	yes|no	yes	Scales energy so that the total energy in ITS-Blocks section is conserved.
ItsBB	scalar	10.	ITS bounding box length multiplier for search.
SierraBB	scalar	10.	SIERRA bounding box length multiplier for search.
integration_point| centroid			Write energy data to either the SIERRA element integration points or centroids. Only one of 'integration_points' or 'centroid' may be specified
end			Ends a set of ITS blocks

Table 4-1. – Input options for the **Its2Sierra** input file. Keywords and default values are given.

A sample input file is given in Input 4.1.

```
// Its2Sierra input file example
Itsblocks = 1 3 7
    mapping = nearest_neighbor
    Sierra_blocks = 2 4 8
    energy_scale_factor = 1
    mesh_scale_factor = 0.0254   #Convert units of ITS mesh
    ItsBB = 20
    SierraBB = 10   #Same as default
    integration_point
end
// Second input section
Itsblocks = 2
    mapping = blockwise
    Sierra_blocks = 3
    energy_scale_factor = 1
    mesh_scale_factor = 0.0254
    centroid
end
```

Input 4.1 – Sample **Its2Sierra** input file

4.2. Output from Its2Sierra

The **Its2Sierra** tool supports output of a new **Exodus** database **output.exo** with the same mesh structure as well as block, sideset, and nodeset names as the input SIERRA mesh file **sierra_mesh.exo**. The file contains a single timestep with data at arbitrary time $t = 0.0$.

4.2.1. Mapped data from ITS

Most importantly, the output mesh file contains element fields representing the mapped `EDEP` in each element from the input **its_output.exo** as well as the computed element volumes. The ITS input **Exodus** file typically contains only a single time step; however, if multiple time steps are present, the `EDEP` data is only mapped from the *final time step* on this file.

The element volume calculated by **Its2Sierra** is always stored as a single value per element on the output file in a field labeled `Volume`, irrespective of whether the 'centroid' or 'integration_points' option is chosen in the input file.

The output `EDEP` data field names and values, however, vary based on the location and mapping strategy chosen in the **Its2Sierra** input file for each ITS block or set of blocks. If the 'centroid' option is chosen, only a single value of `EDEP` is output per element in the output file **output.exo** in an **Exodus** element field labeled "`EDEP`". On the other hand, if the 'integration_points' option is specified, energy deposition data is output at each integration point of each element in the block in an **Exodus** element field labeled `EDEP_<elem_type>_GP<int_pt>`, where `elem_type` represents the element topology of the **Exodus** element block (e.g., `HEX8`, `HEX20`, or `TET4`), and `int_pt` represents the (i, j, k) index of each integration point within the element (e.g., `000`, `100`, `210`). These indices are used internally in **Sierra/SD** to prescribe values correctly at each corresponding element integration point in order to capture gradients in the `EDEP` field.

For example, in an output **Exodus** file containing a block of 20-node hexahedron elements, the fields output from the **Its2Sierra** code are as follows in Output 4.2.

```
Element:  EDEP_elem_HEX20_GP000  EDEP_elem_HEX20_GP001
          EDEP_elem_HEX20_GP002  EDEP_elem_HEX20_GP010
          EDEP_elem_HEX20_GP011  EDEP_elem_HEX20_GP012
          EDEP_elem_HEX20_GP020  EDEP_elem_HEX20_GP021
          EDEP_elem_HEX20_GP022  EDEP_elem_HEX20_GP100
          EDEP_elem_HEX20_GP101  EDEP_elem_HEX20_GP102
          EDEP_elem_HEX20_GP110  EDEP_elem_HEX20_GP111
          EDEP_elem_HEX20_GP112  EDEP_elem_HEX20_GP120
          EDEP_elem_HEX20_GP121  EDEP_elem_HEX20_GP122
          EDEP_elem_HEX20_GP200  EDEP_elem_HEX20_GP201
          EDEP_elem_HEX20_GP202  EDEP_elem_HEX20_GP210
          EDEP_elem_HEX20_GP211  EDEP_elem_HEX20_GP212
          EDEP_elem_HEX20_GP220  EDEP_elem_HEX20_GP221
          EDEP_elem_HEX20_GP222  Volume
```

Output 4.2 – Expected element variables from **Its2Sierra** on a SIERRA mesh of 20-node hexahedron elements.

4.2.2. *Included data from input* SIERRA *and ITS meshes*

If **sierra_mesh.exo** contains preexisting data, including attributes and/or node, element, sideset, or nodeset variables, the **output.exo** file output from **Its2Sierra** contains the same data from *only the final timestep* on **sierra_mesh.exo**.

Any preexisting field data on the input ITS **Exodus** file **its_output.exo** are *not* included in the **output.exo** file. Currently, only data in the EDEP variable may be mapped to the SIERRA mesh output file.

The output **Exodus** file contains any metadata (i.e., QA or information records) associated with both ITS and SIERRA input **Exodus** files, **its_output.exo** and **sierra_mesh.exo**. This includes provenance information from ITS and/or related tools, **Its2Sierra**, **Sierra/SD**, meshing tools, and/or other preprocessing utilities.

5. THEORY

This chapter describes the mapping algorithm theory used by **Its2Sierra**.

5.1. Supported element types

The supported three-dimensional element types for mapping are 4- and 10-noded tetrahedral, eight- and twenty-noded hexahedral, and 6- and 15-noded wedge elements. When the integration_points keyword is used in the mapping parameters in the input file, integration point locations for these elements are derived directly from the default element formulations implemented in **Sierra/SD**. Additional details on default element formulations can be found in the **Sierra/SD** Theory Manual and User Guide.[3,4]

Two-dimensional (shell and membrane) elements are not currently supported, nor are one-dimensional (beam and bar), nor zero-dimensional (sphere and concentrated mass) elements. Blocks with these elements may still exist in the **sierra_mesh.exo** mesh used as input to **Its2Sierra**, but a fatal error occurs if they are mentioned in the input file as **Sierra_Blocks**. These blocks are still output to the **output.exo** mesh, but they do not include any `EDEP` or Volume values.

5.2. Element mapping algorithms

A small selection of mapping strategies is available to enable the user to choose for an appropriate level of fidelity in the mapped energies from the ITS data. Current options include blockwise, nearest neighbor, and inverse-distance-weighted strategies.

5.2.1. Blockwise algorithm

The 'blockwise' algorithm assigns a uniform energy density to the blocks in the SIERRA mesh. As such, it does not preserve *any* gradient variation through each block. Total energy (sum of density $\times$ volume) of all the ITS blocks in the input section is distributed uniformly to the SIERRA blocks listed in the section. The destination blocks in the structural mesh are all assigned the same energy density, determined from their total volume to conserve energy.

5.2.2. The nearest neighbor algorithm

For each SIERRA output point location (centroids or integration points), the 'nearest_neighbor' algorithm finds the nearest ITS data point and maps the energy directly (no interpolation). The nearest neighbor search is done using axis aligned bounding boxes. The length of each side of the bounding boxes defaults to 10 times the cube root of the element volume, or $10\,V^{1/3}$. The factor of 10 can be changed in the input file through the parameters `ItsBB` and/or `SierraBB`.

5.2.3. Inverse-distance-weighted interpolation algorithm

For each SIERRA output point location (centroids or integration points), the 'interpolation' algorithm uses an inverse distance weighting method to map `EDEP` from the **its_output.exo** mesh to the **output.exo** mesh. The energy at each point in the **output.exo** E_s is

$$E_s = \frac{\sum_{i=1}^{N} w_i E_i}{\sum_{i=1}^{N} w_i} \tag{5.2.1}$$

where E_i is the `EDEP` at each ITS point and the weights are defined as inverse distance d between the SIERRA point x_s and the ITS point x_i.

$$w_i = \frac{1}{d\,(x_s, x_i)^2} \tag{5.2.2}$$

For each SIERRA data point, only the ITS points that are both listed in the ItsBlocks section, and lie in the intersection of ItsBoundingBoxes and SierraBoundingBox are considered in the sum. The length of each side of the bounding boxes l_{bbox} defaults to

$$l_{bbox} = 10\sqrt[3]{V_e} \tag{5.2.3}$$

where V_e is the element volume. The factor of 10 can be changed in the input file through the parameters `ItsBB` and/or `SierraBB`.

5.3. Energy conservation

It is important that the total energy of the ITS mesh is conserved when transferred to the SIERRA mesh. The **Conserve_Total_Energy** option computes a scaling factor that is applied to the mapped energy on the SIERRA mesh. **Conserve_Total_Energy** is on by default, and can be turned off with the option **Conserve_Total_Energy** = no. The scaling factor is applied after the mapping is complete to ensure energy conservation. A scaling factor near to unity indicates that the unscaled transfer nearly conserved energy.

The total energy E_{tot} in each **ItsBlocks** section is calculated from the `EDEP` of each element E_e and the volume of each element V_e.

$$E_{tot-ITS} = \sum_{e-ITS} (E_e V_e) \tag{5.3.1}$$

The total energy $E_{tot-Sierra}$ in each **ItsBlocks** section is calculated from the EDEP of each element E_e and the volume of each element V_e.

$$E_{tot-Sierra} = \sum_{e-Sierra} (E_e V_e) \tag{5.3.2}$$

The energy scaling factor F is then

$$F = \frac{E_{tot-ITS}}{E_{tot-Sierra}} \tag{5.3.3}$$

5.4. Diagnostic information

The following section describes the diagnostic output currently available from **Its2Sierra** to diagnose potential issues with mapping EDEP data.

5.4.1. Energy scaling

Energy scaling information (Equation 5.3.3) is written to both standard output and the file 'EnergyScalingStatistics.txt'. If both the ITS mesh and the SIERRA mesh have the same volume, then the energy scale factor should be about 1. If this number is very different from one, it could indicate that something is wrong with your analysis.

Output 5.1 shows a typical output for the 'EnergyScalingStatistics.txt'. In Blocks 101, the SIERRA Hex mesh is a direct refinement of the ITS mesh, so the energy conservation scaling factor is exactly 1.0. For the other (Tet and Wedge) blocks, a scaling factor is applied to make the total energy in each block consistent between the ITS mesh and the SIERRA mesh.

```
Energy scaling factors for each Sierra mesh block for energy conservation:
block 101, energy scaling factor = 1
Energy scaling factors for each Sierra mesh block for energy conservation:
block 102, energy scaling factor = 0.978277
Energy scaling factors for each Sierra mesh block for energy conservation:
block 103, energy scaling factor = 0.999345
Energy scaling factors for each Sierra mesh block for energy conservation:
block 104, energy scaling factor = 1.01638
Energy scaling factors for each Sierra mesh block for energy conservation:
block 105, energy scaling factor = 1
Energy scaling factors for each Sierra mesh block for energy conservation:
block 106, energy scaling factor = 0.999345
```

Output 5.1 – Its2Sierra output showing scale factors for energy.

5.4.2. ITS data point usage

The file ‘itsUsageStatistics.txt’ archives a list of ITS block numbers and element numbers for which the energy input was not used in the mapping to the SIERRA mesh.

Output 5.2 shows the contents of ‘itsUsageStatistics.txt’ when all ITS data points are used in the transfer:

```
All ITS data points were transferred to the Sierra mesh.
```

Output 5.2 – Sample “good” output from **Its2Sierra** when all points in the mesh are matched.

Output 5.3 shows the contents of ’itsUsageStatistics.txt’ when the blocks are not geometrically aligned, so many ITS data points are missed.

```
Number of ITS data points unused in data transfer = 49
The unused blocks and element ids are:

block id = 102, element id = 0 block id = 102, element id = 1 block id = 102,
element id = 2 ...
```

Output 5.3 – Sample “bad” output from **Its2Sierra** when points in the mesh are not matched.

6. BETA MODE

6.1. About Beta Mode

A number of new features are available to users for testing purposes by executing **Its2Sierra** in "beta mode." The following features are enabled by launching **Its2Sierra** in "beta mode:"

- Parallel execution – useful for large meshes
- Source (ITS) and target (SIERRA) mesh transformations (scaling, rotations, and translations)
- Improved interpolation scheme
- User-specified energy deposition field name – enables multiple depositions on a single mesh.

Note that as "beta mode" also deprecates several features, and in some cases features are not yet implemented:

- Mapping command: `Mesh_Scale_Factor` – mesh scaling and transformation now handled outside of `itsblocks`.
- Mapping commands: `ItsBB`, `SierraBB` – default bounding box behaviors are expected to work well enough that user intervention of these low level details is not required.
- Mapping command: `integration_point` – not yet implemented.
- QA Information updates are not yet implemented.
- Preservation of attributes, node numbering, and global variables are not yet tested. Node and element fields are expected to be preserved, however.

6.2. Running Its2Sierra in Parallel

Parallel execution of **Its2Sierra** enables users to rapidly map energy deposition from (or onto) large meshes. The source and target meshes must first be decomposed using a tool such as `decomp` or `stk_balance` first. Then, the **Its2Sierra** program may be executed in parallel using `mpirun` (or similar) command to launch the mapping process:

```
$ decomp --processors=<N> <its_output.e>
$ decomp --processors=<N> <sierra_mesh.g>
$ mpirun -n <N> its2sierra <its_output.e> <sierra_mesh.g> <output.e> <Its2Sierra.inp> --beta
```

Note that the output files of **Its2Sierra** will also be decomposed parallel files, which can be recombined using a tool such as `epu`, or used directly in an application code.

6.3. Mesh transformations

The source (ITS) and target (SIERRA) mesh are often expressed in different coordinate frames or unit systems. To simplify preprocessing requirements, **Its2Sierra** offers a set of mesh transformation commands that may be provided in the input file:

```
// Mesh transformation command syntax
begin mesh modification
  translate [SOURCE|TARGET] mesh in direction [X|Y|Z] <real>(value)

  rotate [SOURCE|TARGET] mesh about [X|Y|Z] AXIS angle = <real>(val_deg) origin =
      ↪ <real>(x0) <real>(y0) <real>(z0)

  scale [SOURCE|TARGET] mesh about origin = <real>(x0) <real>(y0) <real>(z0)
      ↪ scale factor = <real>(value)
end
```

Input 6.1 – Mesh transformation input file syntax

Any number of the `TRANSLATE`, `ROTATE`, or `SCALE` commands may appear in the `BEGIN MESH MODIFICATION` command block. The commands will be executed sequentially as they appear in the input file, which may be important for some transformations that are order-dependent.

Mesh translations may be applied to either the `SOURCE` or `TARGET` mesh. They will be applied along a specified global `X`, `Y`, or `Z` direction.

Mesh rotations may be applied to either the `SOURCE` or `TARGET` mesh. Rotations shall be specified in degrees (not radians), and can be applied about an arbitrary origin given by `x0`, `y0`, and `z0`. All rotations are performed with respect to fixed global coordinate directions.

Mesh scaling transformations may be applied to either the `SOURCE` or `TARGET` mesh. Mesh scaling must specify the origin point given by `x0`, `y0`, and `z0` (which remains stationary) and a scale value.

6.4. Mapping behavior and improved interpolation scheme

"Beta mode" supports the three primary methods of mapping: `nearest_neighbor`, `blockwise`, and `interpolation`.

The `blockwise` mapping is identical to the algorithm described in Section 5.2.1.

The `nearest_neighbor` mapping is almost identical to the algorithm described in Section 5.2.2, with the exception that user-specified bounding boxes are no longer supported. The `nearest_neighbor` algorithm was rewritten to enable parallel execution. Similar to the release version of **Its2Sierra**, the `nearest_neighbor` algorithm uses axis-aligned bounding boxes to

identify overlapping sections of the source and target meshes and assigns values in the target mesh based on the nearest element centroid in source (ITS) mesh. Elements in the target (SIERRA) mesh which are outside of the default search tolerance will automatically perform brute-force search to identify a nearest neighbor. This behavior is transparent to the user and happens automatically without any intervention.

The `interpolation` mapping is new and completely rewritten in "beta mode." The new algorithm produces a better interpolated energy field where high gradients and source/target mesh disparities were problematic for previous interpolation approaches. The `interpolation` mapping scheme works as follows:

1. Element energy density in the source mesh is computed at the nodes by averaging the value of connected elements.
2. Energy is first interpolated to the *nodes* of the target (SIERRA) mesh by identifying the nearest element in the source mesh, which often may entirely encapsulate the target node on the interior of a body. Linear finite element shape functions for the source mesh (currently assumed as HEX8 topology) are used to interpolate energies.
3. Once all nodes in the target mesh have an energy deposition, those values are averaged to obtain *element* values. Future enhancements here may enable interpolation to integration points.
4. An additional option `blend_factor` in the mapping definition may be specified to blend values of a nearest neighbor energy deposition (value = 0, preserves source distribution and gradients better) and an interpolation (value = 1, which provides a smoother energy distribution). The default is currently 0.75.
5. Finally, the resulting energy distribution is smoothed in an attempt to improve the final energy field.

Note that nodes which fall outside of the default bounding box tolerance (nominally the dimension of an element) will be assigned values of the nearest-neighbor element so as to avoid zero values of energy deposition when meshes are not perfectly aligned.

6.5. User-specified energy deposition field name

Users may specify a name for the output element field, which enables multiple sequential executions of **Its2Sierra** from multiple ITS source meshes (which may represent different energy deposition profiles). The input syntax for this is:

```
// Optional: Provide an output variable name
energy variable name = <string>(output_field_name)
```

Input 6.2 – Specify the output energy deposition variable name

This command may be utilized to map multiple energy deposition source (ITS) files as follows:

```
# Decompose all the ITS source meshes onto 4 processors
$ decomp --processors=4 its_output1.e
$ decomp --processors=4 its_output2.e
$ decomp --processors=4 its_output3.e

# Decompose the target mesh onto 4 processors
$ decomp --processors=4 <sierra_mesh.g>

# Perform multiple, chained, mappings:
$ mpirun -n 4 its2sierra its_output1.e <sierra_mesh.g> output1.e  input1.inp --beta
$ mpirun -n 4 its2sierra its_output2.e output1.e  output2.e  input2.inp --beta
$ mpirun -n 4 its2sierra its_output3.e output2.e  final_output.e input3.inp --beta
```

In the above, multiple ITS meshes are decomposed along with a single SIERRA target mesh. There are also multiple input files `input1.inp`, `input2.inp`, `input3.inp` which specify different target variable names such as `EDEP1`, `EDEP2`, `EDEP3` (for example). The first execution of **Its2Sierra** adds the `EDEP1` element field to the target SIERRA mesh and produces an output file `output1.e`. The next execution of **Its2Sierra** adds the `EDEP2` element field by mapping from the `its_output2.e` file and produces the `output2.e` file. Finally, the third execution of **Its2Sierra** adds the `EDEP3` element field by mapping from the `its_output3.e` file and produces the final output `final_output.e`. The file `final_output.e` will have multiple element fields and can be used as input to analysis codes by specifying which element field to read energy deposition data from.

Be aware that when chaining multiple executions of **Its2Sierra** as described here that target mesh modifications should only appear in the first input file (`input1.inp` in the above example) so that they are not executed multiple times, since those target mesh modifications will have already been applied in the output file (`output1.e` in the above example).

7. EXAMPLE

Figure 7-1 depicts an ITS mesh (20-node hexahedrons) with EDEP data. **Its2Sierra** is run with the input file specified in Input 7.1.

```
// Its2Sierra input file for a single ITS block of Hex20 elements.
// The Sierra mesh consists of two blocks: Hex20 elements and Wedge15 elements.
// Nearest neighbor mapping is used, and data is output at the integration points.
Itsblocks = 1
    mapping = nearest_neighbor
    Sierra_blocks = 1 2
    energy_scale_factor = 1
    mesh_scale_factor = 1
    integration_point
end
```

Input 7.1 – Sample **Its2Sierra** input for a model with twenty-node hexahedra.

The SIERRA mesh consists of quadratic wedge and hex elements. The mapped energy is show on the wedge and hex elements in Figures 7-2 and 7-3, respectively.

For further information see the **Sierra/SD** Design Manual.

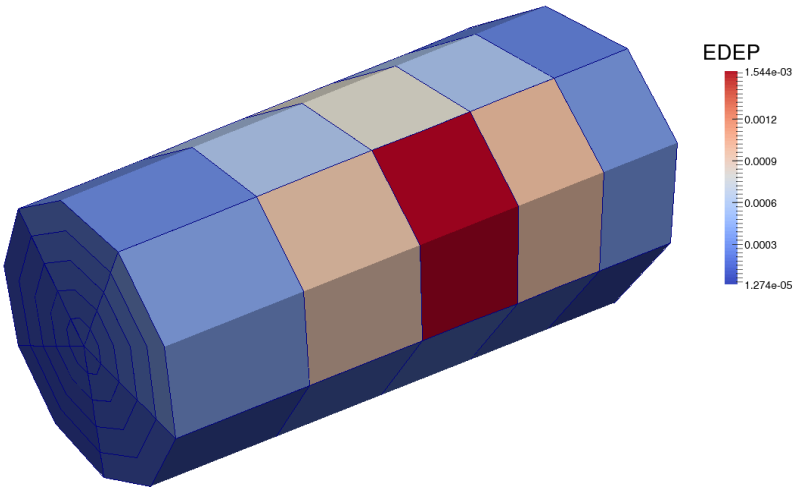

Figure 7-1. – ITS model with EDEP energy.

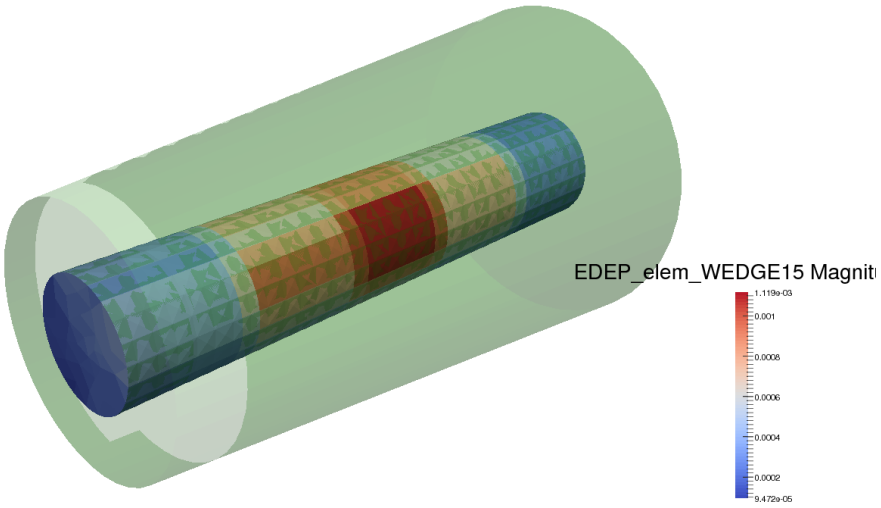

Figure 7-2. – Wedge15 elements in the SIERRA model with mapped ITS energy.

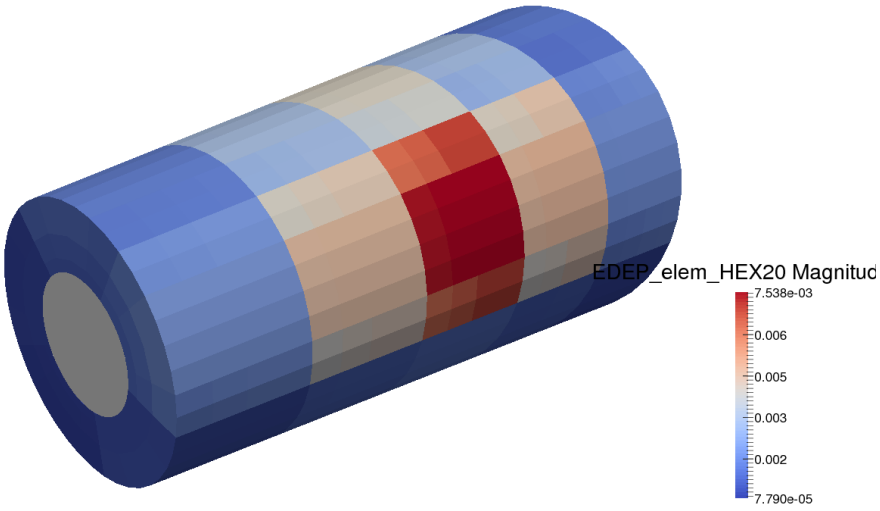

Figure 7-3. – Hex20 elements in the SIERRA model with mapped ITS energy.

This page intentionally left blank.

BIBLIOGRAPHY

[1] Kendall Depriest et al. *ITS Version 6.7: The Integrated TIGER Series of Coupled Electron/Photon Monte Carlo Transport Codes User's Manual*. Tech. rep. SAND2023-12825. Albuquerque, NM: Sandia National Laboratories, Feb. 2023 (cit. on p. 9).

[2] F. Fuentes et al. "Orientation embedded high order shape functions for the exact sequence elements of all shapes". In: *Computers and Mathematics with Applications* 70.1 (2015), pp. 353–458 (cit. on p. 8).

[3] S D Team. *SD – User's Manual*. Tech. rep. SAND2021-12518. living document with more recent versions. PO Box 5800, Albuquerque, NM 87185-5800: Sandia National Laboratories, 2022 (cit. on p. 19).

[4] S D Team. *Sierra Structural Dynamics - Theory Manual*. Tech. rep. SAND2024-01791. PO Box 5800, Albuquerque, NM 87185-5800: Sandia National Laboratory, 2024 (cit. on p. 19).

This page intentionally left blank.

INDEX

This page intentionally left blank.

DISTRIBUTION

Email—Internal

Name	Org.	Sandia Email Address
Technical Library	1911	sanddocs@sandia.gov

Hardcopy—Internal

Number of Copies	Name	Org.	Mailstop
1	K. H. Pierson	1542	0845

This page intentionally left blank.

Sandia National Laboratories is a multimission laboratory managed and operated by National Technology & Engineering Solutions of Sandia LLC, a wholly owned subsidiary of Honeywell International Inc., for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-NA0003525.