Computational Methods in Ice Sheet Modeling for Next-Generation Climate Simulations

Irina K. Tezaur, et al.

Quantitative Modeling and Analysis Department
Sandia National Laboratories, Livermore, CA

AME Seminar                  University of Notre Dame            February 23, 2021

SAND2021-1606P

Computing resources: NERSC, OLCF.

Support for this work was provided through Scientific Discovery through Advanced Computing (SciDAC) projects funded by the U.S. Department of Energy, Office of Science, Advanced Scientific Computing Research (ASCR) and Biological and Environmental Research (BER) → SciDAC Application Partnership.
Outline

1. Background
   • Motivation for climate & land-ice modeling
   • ISMs, ESMs & projects
   • Land-ice equations
   • Our codes: ALI, MALI

2. Algorithms and software
   • Discretization & meshes
   • Nonlinear solvers
   • Linear solvers
   • Performance-portability
   • Ice sheet initialization
   • Towards UQ

3. Simulations

4. Ongoing & future work
Outline

1. Background
   • Motivation for climate & land-ice modeling
   • ISMs, ESMs & projects
   • Land-ice equations
   • Our codes: ALI, MALI

2. Algorithms and software
   • Discretization & meshes
   • Nonlinear solvers
   • Linear solvers
   • Performance-portability
   • Ice sheet initialization
   • Towards UQ

3. Simulations

4. Ongoing & future work
Motivation

➢ **Climate change** has been declared a “**national security issue**” by President Joe Biden.

➢ Global mean sea-level is rising at the rate of **3.2 mm/year** and this rate is **increasing**, with the latest studies suggesting a possible increase in sea-level of **0.3-2.5 m** by 2100.

❖ Due to **melting of the polar ice sheets** (Greenland, Antarctica).

➢ **Full deglaciation**: sea level could rise up to ~65 m (Antarctica: 58 m, Greenland: 7 m)

*Estimates given by Prof. Richard Alley of Penn State.
Motivation

➢ Climate change has been declared a “national security issue” by President Joe Biden.
➢ Global mean sea-level is rising at the rate of 3.2 mm/year and this rate is increasing, with the latest studies suggesting a possible increase in sea-level of 0.3-2.5 m by 2100.
❖ Due to melting of the polar ice sheets (Greenland, Antarctica).
➢ Full deglaciation*: sea level could rise up to ~65 m (Antarctica: 58 m, Greenland: 7 m)

*Estimates given by Prof. Richard Alley of Penn State.

Map of North America showing 6 m SLR (NASA)  
Total mass loss of ice sheets b/w 1992-2011

Modeling of ice sheet dynamics is essential for providing estimates of sea-level rise, towards understanding the local/global effects of climate change.
1. Background
   • Motivation for climate & land-ice modeling
   • ISMs, ESMs & projects
   • Land-ice equations
   • Our codes: ALI, MALI

2. Algorithms and software
   • Discretization & meshes
   • Nonlinear solvers
   • Linear solvers
   • Performance-portability
   • Ice sheet initialization
   • Towards UQ

3. Simulations

4. Ongoing & future work
What is an Ice Sheet Model (ISM)?

**Dynamical core (”dycore”)**
Conservation of:
- Mass (ice thickness)
- Momentum (ice velocity)
- Energy (ice temperature)

**Physical processes (”physics”)**
- Iceberg calving
- Basal sliding
- Etc...

**Climate forcing**
- Snowfall/melt
- Ocean melting/freezing
- Etc...
What is an Ice Sheet Model (ISM)?

**Dynamical core ("dycore")**
Conservation of:
- Mass (ice thickness)
- Momentum (ice velocity)
- Energy (ice temperature)

**Physical processes ("physics")**
- Iceberg calving
- Basal sliding
- Etc...

**Climate forcing**
- Snowfall/melt
- Ocean melting/freezing
- Etc...

Comes from *Earth System Model (ESM)*
Earth System Models (ESMs)

An Earth System Model (ESM) has **six modular components**: 

- Atmos. (EAM)
- Ocean (MPAS-O)
- Sea Ice (MPAS-SI)
- Land (ALM)
- Land Ice (MALI)
- Flux Coupler

**CESM**
COMMUNITY EARTH SYSTEM MODEL
Earth System Models (ESMs)

An Earth System Model (ESM) has six modular components:

- Atmos. (EAM)
- Ocean (MPAS-O)
- Sea Ice (MPAS-SI)
- Land (ALM)
- Land Ice (MALI)

**Goal of ESM:** to provide actionable scientific predictions of 21st century sea-level change (including uncertainty bounds).
An Earth System Model (ESM) has **six modular components**:

- Atmos. (EAM)
- Ocean (MPAS-O)
- Sea Ice (MPAS-SI)
- Land (ALM)
- Land Ice (MALI)
- Flux Coupler

**Goal of ESM**: to provide actionable scientific predictions of 21st century sea-level change (including uncertainty bounds).

About a decade ago, existing land-ice models were **not robust enough** for ESM integration! 😞
U.S. DOE Ice Sheet/Climate Model Efforts

Motivation:

• 2007 IPCC (Intergovernmental Panel on Climate Change) Fourth Assessment Report *declined* to include estimates of future sea-level rise from ice sheet dynamics due to the *inability* of ice sheet models to mimic/explain observed dynamic behaviors.

➢ “*Much work is needed* to make [present-day ISMs] robust and efficient on continental scales and to quantify uncertainties in their projected outputs”. – IPCC AR4 (2007)
U.S. DOE Ice Sheet/Climate Model Efforts

Motivation:

• 2007 IPCC (Intergovernmental Panel on Climate Change) Fourth Assessment Report declined to include estimates of future sea-level rise from ice sheet dynamics due to the inability of ice sheet models to mimic/explain observed dynamic behaviors.

  ➢ “Much work is needed to make [present-day ISMs] robust and efficient on continental scales and to quantify uncertainties in their projected outputs”. – IPCC AR4 (2007)

DOE-funded Land-Ice Modeling Projects:

• Probabilistic Sea-Level Projections from Ice Sheet Models and ESMs (ProSPect): 2017-2022.

Aim is to develop & apply robust, accurate, scalable dynamical cores for ice sheet modeling on unstructured meshes, enable uncertainty quantification (UQ), and integrate models/tools into DOE E3SM
U.S. DOE Ice Sheet/Climate Model Efforts

Motivation:

- 2007 IPCC (Intergovernmental Panel on Climate Change) Fourth Assessment Report *declined* to include estimates of future sea-level rise from ice sheet dynamics due to the *inability* of ice sheet models to mimic/explain observed dynamic behaviors.

  ➢ “Much work is needed to make [present-day ISMs] robust and efficient on continental scales and to quantify uncertainties in their projected outputs”. – IPCC AR4 (2007)

DOE-funded Land-Ice Modeling Projects:

- *Probabilistic Sea-Level Projections from Ice Sheet Models and ESMs (ProSPect)*: 2017-2022.

Aim is to *develop & apply robust, accurate, scalable* dynamical cores for ice sheet modeling on *unstructured* meshes, enable *uncertainty quantification* (UQ), and *integrate* models/tools into DOE E3SM

DOE Energy Exascale Earth System Model (E3SM):

- “Next-generation” climate model with focus of *decadal-century timescale projections, high-spatial resolution*, next *generation HPC*, impacts to *U.S. infrastructure*. 
The PISCEES & ProSPect Projects

**PISCEES (2012-2017)**  
ProSPect (2017-present)  
*SciDAC Application Partnerships*  
*(DOE’s BER + ASCR divisions)*

Two land-ice dycores currently under development

**MALI**  
*Sandia National Labs*  
Finite Element “First Order” Stokes Model

**BISICLES**  
*Lawrence Berkeley National Lab*  
Finite Volume + AMR L1L2 Model

**MALI**: MPAS-Albany Land Ice  
**BISICLES**: Berkeley Ice Sheet Initiative for Climate at Extreme Scales
The PISCEES & ProSPect Projects

PISCEES (2012-2017)
ProSPect (2017-present)
SciDAC Application Partnerships (DOE’s BER + ASCR divisions)

Two land-ice dycores currently under development

MALI
Sandia National Labs
Finite Element “First Order” Stokes Model

BISICLES
Lawrence Berkeley National Lab
Finite Volume + AMR L1L2 Model

MALI: MPAS-Albany Land Ice
BISICLES: Berkeley Ice Sheet Initiative for Climate at Extreme Scales
Outline

1. Background
   • Motivation for climate & land-ice modeling
   • ISMs, ESMs & projects
   • **Land-ice equations**
   • Our codes: ALI, MALI

2. Algorithms and software
   • Discretization & meshes
   • Nonlinear solvers
   • Linear solvers
   • Performance-portability
   • Ice sheet initialization
   • Towards UQ

3. Simulations

4. Ongoing & future work
Stokes Ice Flow Equations

Ice behaves like a very viscous non-Newtonian shear-thinning fluid (like lava flow) and is modeled quasi-statically using nonlinear incompressible Stokes equations.

\[ \begin{align*}
-\nabla \cdot \tau + \nabla p &= \rho g, \\
\nabla \cdot \mathbf{u} &= 0
\end{align*} \], in \( \Omega \)

- Fluid velocity vector: \( \mathbf{u} = (u_1, u_2, u_3) \)
- Isotropic ice pressure: \( p \)
- Deviatoric stress tensor: \( \tau = 2\mu \varepsilon \)
- Strain rate tensor: \( \varepsilon_{ij} = \frac{1}{2} \left( \frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i} \right) \)
- Glen’s Law Viscosity*: \( \mu = \frac{1}{2} A(T)^{-\frac{1}{n}} \left( \frac{1}{2} \sum_{ij} \varepsilon_{ij}^2 \right)^{\left( \frac{1}{2n} - \frac{1}{2} \right)} \)
- Flow factor: \( A(T) = A_0 e^{-\frac{Q}{RT}} \)

*Nye 1957; Cuffey et al., 2010. Typically we use \( n = 3 \).
Stokes Ice Flow Equations

Ice behaves like a very viscous non-Newtonian shear-thinning fluid (like lava flow) and is modeled quasi-statically using nonlinear incompressible Stokes equations.

\[
\begin{aligned}
- \nabla \cdot \tau + \nabla p &= \rho g, \text{ in } \Omega \\
\nabla \cdot \mathbf{u} &= 0 \\
\end{aligned}
\]

- Fluid velocity vector: \( \mathbf{u} = (u_1, u_2, u_3) \)
- Isotropic ice pressure: \( p \)
- Deviatoric stress tensor: \( \tau = 2\mu \boldsymbol{\varepsilon} \)
- Strain rate tensor: \( \varepsilon_{ij} = \frac{1}{2} \left( \frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i} \right) \)
- Glen’s Law Viscosity*: \( \mu = \frac{1}{2} A(T)^{-\frac{1}{n}} \left( \frac{1}{2} \sum_{ij} \varepsilon_{ij}^2 \right)^{\left( \frac{1}{2n} - \frac{1}{2} \right)} \)
- Flow factor: \( A(T) = A_0 e^{-\frac{Q}{RT}} \)

*Highly nonlinear rheology!

*Nye 1957; Cuffey et al., 2010. Typically we use \( n = 3 \).
Stokes Ice Flow Equations

Ice behaves like a very viscous non-Newtonian shear-thinning fluid (like lava flow) and is modeled quasi-statically using nonlinear incompressible Stokes equations.

\[\begin{align*}
-\nabla \cdot \tau + \nabla p &= \rho g, \quad \text{in } \Omega \\
\nabla \cdot \mathbf{u} &= 0
\end{align*}\]

- Fluid velocity vector: \( \mathbf{u} = (u_1, u_2, u_3) \)
- Isotropic ice pressure: \( p \)
- Deviatoric stress tensor: \( \tau = 2\mu\varepsilon \)
- Strain rate tensor: \( \varepsilon_{ij} = \frac{1}{2} \left( \frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i} \right) \)
- Glen’s Law Viscosity*: \( \mu = \frac{1}{2} A(T)^{-\frac{1}{n}} \left( \frac{1}{2} \sum_{ij} \varepsilon_{ij}^2 \right)^{\left(\frac{1}{2n} - \frac{1}{2}\right)} \)
- Flow factor: \( A(T) = A_0 e^{-\frac{Q}{RT}} \) "Gold standard" model

* Nye 1957; Cuffey et al., 2010. Typically we use \( n = 3 \).
Stokes Ice Flow Equations

Ice behaves like a **very viscous non-Newtonian shear-thinning fluid** (like lava flow) and is modeled **quasi-statically** using **nonlinear incompressible Stokes equations**.

\[
\begin{align*}
- \nabla \cdot \tau + \nabla p &= \rho g, \\
\nabla \cdot \mathbf{u} &= 0,
\end{align*}
\]

in \( \Omega \)

- Fluid velocity vector: \( \mathbf{u} = (u_1, u_2, u_3) \)
- Isotropic ice pressure: \( p \)
- Deviatoric stress tensor: \( \tau = 2\mu \varepsilon \)
- Strain rate tensor: \( \varepsilon_{ij} = \frac{1}{2} \left( \frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i} \right) \)
- Glen’s Law Viscosity*: \( \mu = \frac{1}{2} A(T) \left( \frac{1}{2} \sum_{ij} \varepsilon_{ij}^2 \right)^{\frac{1}{2n}} \)
- Flow factor: \( A(T) = A_0 e^{-\frac{Q}{RT}} \) **Highly nonlinear rheology!**

*Nye 1957; Cuffey et al., 2010. Typically we use \( n = 3 \).
First Order (FO) Stokes/Blatter-Pattyn Model*

Stokes($\mathbf{u}, p$) in $\Omega \in \mathbb{R}^3$

$$\mathbf{u} \equiv (u, v, w)$$

$$\varepsilon(\mathbf{u}) = \begin{pmatrix}
    u_x & \frac{1}{2}(u_y + v_x) & \frac{1}{2}(u_z + w_x) \\
    \frac{1}{2}(u_y + v_x) & v_y & \frac{1}{2}(v_z + w_y) \\
    \frac{1}{2}(u_z + w_x) & \frac{1}{2}(v_z + w_y) & w_z
\end{pmatrix}$$

$$p = \rho g(s - z) - 2\mu(u_x + v_y)$$

*Pattyn, 2003; Blatter, 1995.
First Order (FO) Stokes/Blatter-Pattyn Model*

Hydrostatic approximation + scaling argument based on the fact that ice sheets are thin and normals are almost vertical

First Order Stokes (a.k.a. Blatter-Pattyn) Model

Stokes \((\mathbf{u}, p)\) in \(\Omega \in \mathbb{R}^3\)

\[
\epsilon(\mathbf{u}) = \begin{pmatrix}
  u_x & \frac{1}{2} (u_y + v_x) & \frac{1}{2} (u_z + w_x) \\
  \frac{1}{2} (u_y + v_x) & v_y & \frac{1}{2} (v_z + w_y) \\
  \frac{1}{2} (u_z + w_x) & \frac{1}{2} (v_z + w_y) & w_z 
\end{pmatrix}
\]

\[p = \rho g (s - z) - 2\mu (u_x + v)\]

\[
\dot{\epsilon}(\mathbf{u}, v) = \begin{pmatrix}
  2u_x + v_y & \frac{1}{2} (u_y + v_x) & \frac{1}{2} u_z \\
  \frac{1}{2} (u_y + v_x) & u_x + 2v_y & \frac{1}{2} v_z 
\end{pmatrix}
\]

\[
\mu = \frac{1}{2} A(T)^{-\frac{1}{n}} \left( \frac{1}{2} \sum_{ij} \dot{\epsilon}_{ij}^2 \right)^{\frac{1}{2n-1}}
\]

Discussion:

- Nice “elliptic” approximation to full Stokes.
- 3D model for two unknowns \((u, v)\) with nonlinear \(\mu\).
- Valid for both Greenland and Antarctica and used in continental scale simulations.

*Pattyn, 2003; Blatter, 1995.
Boundary Conditions

Ice-Atmosphere Boundary:

- **Stress-free BC:** \(2\mu \dot{e}_i \cdot n = 0\) on \(\Gamma_s\)

Ice-Bedrock Boundary:

- **Basal sliding BC:** \(2\mu \dot{e}_i \cdot n + \beta u_i = 0\) on \(\Gamma_\beta\)

\[
\beta = \text{basal sliding coefficient} \\
\beta = \beta(x, y) \text{ or } \beta = \beta(x, y, u, t)
\]

Can’t be measured – must be estimated from data!

Ice-Ocean Boundary:

- **Floating ice (a.k.a. open ocean) BC:**

\[
2\mu \dot{e}_i \cdot n = \begin{cases} 
\rho g z n, & \text{if } z > 0 \\
0, & \text{if } z \leq 0
\end{cases}
\] on \(\Gamma_l\)

IPCC WG1 (2013): “Based on current understanding, only the collapse of marine-based sectors of the Antarctic ice sheet, if initiated, could cause [SLR by 2100] substantially above the likely range [of ~0.5-1 m].”

Boundary conditions have tremendous effect on ice sheet behavior!
Ice Sheet Evolution

Ice velocity equations are **coupled** with equations for ice sheet evolution (thickness) and ice temperature.

- **Energy equation** for the temperature $T$:
  \[
  \rho c \frac{\partial T}{\partial t} + \rho c u \cdot \nabla T = \nabla \cdot (k \nabla T) + 2 \dot{\epsilon} \sigma, \quad \text{in } \Omega_H
  \]
  - Flow factor $A$ in Glen’s law viscosity $\mu$ is function of $T$.

- **Mass equation** for the ice thickness $H$:
  \[
  \frac{\partial H}{\partial t} + \nabla \cdot (\bar{u}H) + \dot{b}, \quad \text{on } \Gamma
  \]
  - $\bar{u} = \text{vertically averaged } u$
  - $\dot{b} = \text{surface mass balance}$
  - (given accumulation-ablation function that accounts for e.g. accumulation due to snowfall)
  - $\Gamma = \text{horizontal extent of the ice}$

- Thickness $H$ determines the **geometry** for velocity equations.

- Ice-covered (“active”) cells shaded in white ($H > H_{\text{min}}$)
Ice Sheet Evolution

Ice velocity equations are **coupled** with equations for ice sheet evolution (thickness) and ice temperature.

- **Energy equation** for the temperature $T$:

$$\rho c \frac{\partial T}{\partial t} + \rho c \mathbf{u} \cdot \nabla T = \nabla \cdot (k \nabla T) + 2 \dot{\varepsilon} \sigma, \quad \text{in } \Omega_H$$

- Flow factor $A$ in Glen’s law viscosity $\mu$ is function of $T$.

- **Mass equation** for the ice thickness $H$:

$$\frac{\partial H}{\partial t} + \nabla \cdot (\overline{\mathbf{u}} H) + \dot{b}, \quad \text{on } \Gamma$$

  - $\overline{\mathbf{u}} =$ vertically averaged $\mathbf{u}$
  - $\dot{b} =$ surface mass balance
  - (given accumulation-ablation function that accounts for e.g. accumulation due to snowfall)
  - $\Gamma =$ horizontal extent of the ice

- Thickness $H$ determines the **geometry** for velocity equations.

- Ice-covered (“active”) cells shaded in white ($H > H_{\text{min}}$)
Ice Sheet Evolution

Ice velocity equations are **coupled** with equations for ice sheet evolution (thickness) and ice temperature.

- **Energy equation** for the temperature $T$:

  \[ \rho c \frac{\partial T}{\partial t} + \rho c \mathbf{u} \cdot \nabla T = \nabla \cdot (k \nabla T) + 2 \dot{\varepsilon} \sigma, \quad \text{in} \quad \Omega_H \]

  ➢ Flow factor $A$ in Glen’s law viscosity $\mu$ is function of $T$.

- **Mass equation** for the ice thickness $H$:

  \[ \frac{\partial H}{\partial t} + \nabla \cdot (\overline{\mathbf{u}}H) + \dot{b}, \quad \text{on} \quad \Gamma \]

  - $\overline{\mathbf{u}} = \text{vertically averaged } \mathbf{u}$
  - $\dot{b} = \text{surface mass balance}$
  - *(given accumulation-ablation function that accounts for e.g. accumulation due to snowfall)*
  - $\Gamma = \text{horizontal extent of the ice}$

  ➢ Thickness $H$ determines the **geometry** for velocity equations.

**Ice-covered (“active”) cells shaded in white**

$(H > H_{\text{min}})$
Outline

1. Background
   • Motivation for climate & land-ice modeling
   • ISMs, ESMs & projects
   • Land-ice equations
   • Our codes: ALI, MALI

2. Algorithms and software
   • Discretization & meshes
   • Nonlinear solvers
   • Linear solvers
   • Performance-portability
   • Ice sheet initialization
   • Towards UQ

3. Simulations

4. Ongoing & future work
Our Codes

Momentum Balance: **First-Order Stokes** PDEs

\[
\begin{align*}
-\nabla \cdot (2\mu \dot{\varepsilon}_1) &= -\rho g \frac{\partial s}{\partial x}, \text{ in } \Omega \\
-\nabla \cdot (2\mu \dot{\varepsilon}_2) &= -\rho g \frac{\partial s}{\partial y}
\end{align*}
\]

with **Glen’s law** viscosity \( \mu = \frac{1}{2} A(T)^{-\frac{1}{3}} \left( \frac{1}{2} \sum_{ij} \dot{\varepsilon}_{ij}^2 \right)^{\frac{2}{3}} \).

Energy Balance: **temperature** advection-diffusion PDE

\[
\rho c \frac{\partial T}{\partial t} = \nabla \cdot (k\nabla T) - \rho c \mathbf{u} \cdot \nabla T + 2\dot{\varepsilon} \sigma
\]

Conservation of Mass: **thickness** evolution PDE

\[
\frac{\partial H}{\partial t} = -\nabla \cdot (\bar{\mathbf{u}} H) + \dot{b}
\]

*Codes:*

**Albany** = multi-physics PDE code*

**Albany Land-Ice (ALI)**

**MPAS** Model for Prediction Across Scales

**E3SM** Energy Exascale Earth System Model

*https://github.com/SNLComputation/Albany.*
**Our Codes**

**Velocity solve is most expensive!**

**Momentum Balance:** *First-Order Stokes* PDEs

\[
\begin{align*}
-\nabla \cdot (2\mu \dot{\varepsilon}_1) &= -\rho g \frac{\partial s}{\partial x}, \quad \text{in } \Omega \\
-\nabla \cdot (2\mu \dot{\varepsilon}_2) &= -\rho g \frac{\partial s}{\partial y}
\end{align*}
\]

with *Glen’s law* viscosity \(\mu = \frac{1}{2} A(T) \left(\frac{1}{3} \left(\frac{1}{2} \sum_{ij} \dot{\varepsilon}_{ij}^2 \right)^{-\frac{2}{3}}\right)\).

This talk will focus on the velocity solve.

**Energy Balance:** *temperature* advection-diffusion PDE

\[
\rho c \frac{\partial T}{\partial t} = \nabla \cdot (k \nabla T) - \rho c \mathbf{u} \cdot \nabla T + 2\dot{\varepsilon} \sigma
\]

**Conservation of Mass:** *thickness* evolution PDE

\[
\frac{\partial H}{\partial t} = -\nabla \cdot (\mathbf{u} H) + b
\]

**Codes:**

**Albany** = multi-physics PDE code*

**Albany Land-Ice (ALI)**

**Moshi**

**MPAS**

**Model for Prediction Across Scales**

---

*https://github.com/SNLComputation/Albany.
Albany Land-Ice (ALI) FO Stokes Solver

The **Albany Land-Ice** First Order Stokes solver is implemented in a Sandia open-source parallel C++ multi-physics finite element code known as...

**Agile Components**
- Discretizations/meshes
- Solver libraries
- Preconditioners
- Automatic differentiation
- Performance portable kernels
- Many others!

- Parameter estimation
- Uncertainty quantification
- Optimization
- Bayesian inference

**Albany:**
https://github.com/SNL Computation/Albany

**Trilinos:**
https://github.com/trilinos/Trilinos

**Dakota:**
https://dakota.sandia.gov/
Model for Prediction Across Scales (MPAS): climate modeling framework built around SCVT* meshes (LANL + NCAR collaboration)

*SCVT = Spherical Centroidal Voronoi Tesselations

- Ocean\(^1\), sea ice\(^2\), and land ice\(^3\) dynamical cores
- Built using shared software framework
- New capabilities added to one core benefit all others

\(^1\) Ringler et al., 2013; \(^2\) Turner et al. (in prep); \(^3\) Hoffman et al. (in prep)
MPAS + ALI Coupling (MALI)

MPAS Land-Ice (Fortran)
Thickness evolution, temperature solve, coupling to DOE-ESM

LandIce_model

output file

MPAS Land-Ice (Fortran)

C++/Fortran Interface, Mesh Conversion

Albany Land-Ice (C++)
velocity solve

\[ \frac{\partial H}{\partial t} = -\nabla \cdot (\mathbf{u}H) + \dot{b} \]

\[ \rho c \frac{\partial T}{\partial t} = \nabla \cdot (k\nabla T) - \rho c \mathbf{u} \cdot \nabla T + 2\dot{e}\sigma \]

\[
\begin{align*}
-\nabla \cdot (2\mu \dot{e}_1) &= -\rho g \frac{\partial s}{\partial x} \\
-\nabla \cdot (2\mu \dot{e}_2) &= -\rho g \frac{\partial s}{\partial y}
\end{align*}
\]

“Loose” sequential/staggered coupling between \( \mathbf{u} \) and \((T, H)\).

- Making this coupling **tighter** by moving thickness and temperature evolution to Albany is WIP.
Outline

1. Background
   • Motivation for climate & land-ice modeling
   • ISMs, ESMs & projects
   • Land-ice equations
   • Our codes: ALI, MALI

2. Algorithms and software
   • Discretization & meshes
   • Nonlinear solvers
   • Linear solvers
   • Performance-portability
   • Ice sheet initialization
   • Towards UQ

3. Simulations

4. Ongoing & future work
Finite Element Discretization

- Can handle well the *boundary conditions* arising in land ice modeling.
- Allow the use of *unstructured meshes* to concentrate the computational power where it is needed.

Greenland mesh from ALI refined based on gradient of surface velocity
Meshes

- ALI runs employ dual of hexagonal mesh from MPAS extruded to tetrahedra for the velocity solve in Albany.

- Meshes are structured (extruded) in the vertical dimension.

- Ice sheets are thin (thickness up to 4 km, horizontal extension of thousands km), meaning we typically have elements with bad aspect ratios.
Outline

1. Background
   • Motivation for climate & land-ice modeling
   • ISMs, ESMs & projects
   • Land-ice equations
   • Our codes: ALI, MALI

2. Algorithms and software
   • Discretization & meshes
   • **Nonlinear solvers**
   • Linear solvers
   • Velocity-temperature coupling
   • Performance-portability
   • Ice sheet initialization
   • Towards UQ

3. Simulations

4. Ongoing & future work
Nonlinear Solver for Discretized Problem

- *Picard iterations* have been method of choice in ice sheet modeling
Nonlinear Solver for Discretized Problem

- **Picard iterations** have been method of choice in ice sheet modeling
- ALI employs **Newton’s method** with several advancements:
Nonlinear Solver for Discretized Problem

- **Picard iterations** have been method of choice in ice sheet modeling
- ALI employs **Newton's method** with several advancements:
  - **Automatic differentiation (AD)** Jacobian – gives you exact derivatives/Jacobians without deriving/hand-coding them!

Libraries (Sacado) provides new scalar types that *overload the math operators* to propagate embedded quantities via chain rule

- **Derivatives**: DFad<double>
- **Hessians**: DFad<SFad<double,N>>
- **Stochastic Galerkin resid**: PCE<double>
- **Stochastic Galerkin Jac**: DFad<PCE<double>
- **Sensitivites**: DFad<double>

<table>
<thead>
<tr>
<th>double</th>
<th>DFad&lt;double&gt;</th>
</tr>
</thead>
<tbody>
<tr>
<td>Operation</td>
<td>Overloaded AD impl</td>
</tr>
<tr>
<td>(c = a \pm b)</td>
<td>(\dot{c} = \dot{a} \pm \dot{b})</td>
</tr>
<tr>
<td>(c = ab)</td>
<td>(\dot{c} = \dot{a} \cdot \dot{b} + \dot{b} \cdot \dot{a})</td>
</tr>
<tr>
<td>(c = a/b)</td>
<td>(\dot{c} = \frac{\dot{a} - \dot{b} \cdot \dot{a}}{\dot{b}})</td>
</tr>
<tr>
<td>(c = a^r)</td>
<td>(\dot{c} = ra^{r-1})</td>
</tr>
</tbody>
</table>

*Tezaur et al. 2015.*

No finite difference truncation error!
Nonlinear Solver for Discretized Problem

- **Picard iterations** have been method of choice in ice sheet modeling
- ALI employs **Newton’s method** with several advancements:
  - **Automatic differentiation (AD) Jacobian** – gives you exact derivatives/Jacobians without deriving/hand-coding them!
  - **Homotopy continuation*** to deal with “singular” viscosity.

*Tezaur et al. 2015.*
Nonlinear Solver for Discretized Problem

- **Picard iterations** have been method of choice in ice sheet modeling
- ALI employs **Newton’s method** with several advancements:
  - **Automatic differentiation (AD)** Jacobian – gives you exact derivatives/Jacobians without deriving/hand-coding them!
  - **Homotopy continuation*** to deal with “singular” viscosity.

**Glen’s Law Viscosity:**

\[
\mu = \frac{1}{2} A(T)^{-\frac{1}{3}} \left( \frac{1}{2} \sum_{ij} \dot{\varepsilon}_{ij}^2 \right)^{-2/3}
\]

*Undefined for \( u = \text{const} \!*

**Nonlinear Solver for Discretized Problem**

\[
\dot{\varepsilon}_{ij} = \frac{1}{2} \left( \frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i} \right)
\]

*Tezaur et al. 2015.*
Nonlinear Solver for Discretized Problem

• **Picard iterations** have been method of choice in ice sheet modeling
• ALI employs **Newton’s method** with several advancements:
  ➢ **Automatic differentiation (AD)** Jacobian – gives you exact derivatives/Jacobians without deriving/hand-coding them!
  ➢ **Homotopy continuation*** to deal with “singular” viscosity.

Glen’s Law Viscosity:

\[
\mu = \frac{1}{2} A(T)^{-\frac{1}{3}} \left( \frac{1}{2} \sum_{ij} \dot{\varepsilon}_{ij}^2 + \gamma \right)^{-2/3}
\]

\[
\gamma = \text{regularization parameter (} O(1e-10) \text{)}
\]

\[
\dot{\varepsilon}_{ij} = \frac{1}{2} \left( \frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i} \right)
\]

*Tezaur et al. 2015.*
Nonlinear Solver for Discretized Problem

- **Picard iterations** have been method of choice in ice sheet modeling
- ALI employs **Newton’s method** with several advancements:
  - **Automatic differentiation (AD) Jacobian** – gives you exact derivatives/Jacobians without deriving/hand-coding them!
  - **Homotopy continuation*** to deal with “singular” viscosity.

\[
\mu = \frac{1}{2} A(T)^{-\frac{1}{3}} \left( \frac{1}{2} \sum_{ij} \dot{\varepsilon}_{ij}^2 + \gamma \right)^{-\frac{2}{3}}
\]

\[
\gamma = \text{regularization parameter}
\]

\[
\dot{\varepsilon}_{ij} = \frac{1}{2} \left( \frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i} \right)
\]

Improvised **robustness** and **faster** nonlinear convergence by doing a **homotopy continuation** w.r.t. \(\gamma\)

*Tezaur et al. 2015.*
Outline

1. Background
   • Motivation for climate & land-ice modeling
   • ISMs, ESMs & projects
   • Land-ice equations
   • Our codes: ALI, MALI

2. Algorithms and software
   • Discretization & meshes
   • Nonlinear solvers
   • Linear solvers
   • Performance-portability
   • Ice sheet initialization
   • Towards UQ

3. Simulations

4. Ongoing & future work
From Nonlinear Solvers to Linear Solvers

- **Krylov iterative linear solvers** are employed – CG or GMRES.
  - FO Stokes equations are symmetric.
- Grid partitioning is done on **2D base grid** for best linear solver performance (recall that mesh is layered).

- **Bad aspect ratios, floating ice, and island/ice hinges** can wreak havoc on linear solver!
  - Specialized **solvers/preconditioners** have been developed in Trilinos to deal w/ these issues.
    - AMG\(^1\) preconditioner w/ semi-coarsening\(^2\).
    - Fast and Robust Overlapping Schwarz (FROSch) preconditioner\(^3\) w/ GDSW\(^4\) coarse spaces
  - **Graph-based algorithms** for removing islands/ice hinges are being developed\(^2\).

---

1 Algebraic Multi-Grid. 2 Tuminaro et al. 2016. 3 Heinlein et al. 2020. 4 Generalized-Dryja-Smith-Widlund.
From Nonlinear Solvers to Linear Solvers

- **Krylov iterative linear solvers** are employed – CG or GMRES.
  - FO Stokes equations are symmetric.
- Grid partitioning is done on **2D base grid** for best linear solver performance (recall that mesh is layered).

- **Bad aspect ratios, floating ice, and island/ice hinges** can **wreak havoc** on linear solver!

  - Specialized **solvers/preconditioners** have been developed in Trilinos to deal w/ these issues.
    - AMG\(^1\) preconditioner w/ semi-coarsening\(^2\).
    - Fast and Robust Overlapping Schwarz (FROSch) preconditioner\(^3\) w/ GDSW\(^4\) coarse spaces
  - **Graph-based algorithms** for removing islands/ice hinges are being developed\(^2\).

---

How Does Multi-Grid Work?

**Basic idea:** accelerate convergence of an iterative method on a given grid by solving a series of (cheaper) problems on coarser grids.

- Create set of **coarse approximations**.
- Apply **restriction operator** $R_i$ to interpolate from fine to coarse grid.
- **Solve** problem on coarse grid.
- Apply **prolongation operator** $P_i$ to get back to original (fine) grid.
- **Smoothers** are applied throughout procedure to reduce short wavelength errors.

**Solve** $A_3 u_3 = f_3$

Smooth $A_3 u_3 = f_3$. Set $f_2 = R_2 r_3$.

Smooth $A_2 u_2 = f_2$. Set $f_1 = R_1 r_2$.

Solve $A_1 u_1 = f_1$ directly.

Set $u_3 = u_3 + P_2 u_2$. Smooth $A_3 u_3 = f_3$.

Set $u_2 = u_2 + P_1 u_1$. Smooth $A_2 u_2 = f_2$. 

Scalable Algebraic Multi-Grid (AMG) Preconditioners

**Bad aspect ratios** \((dx \gg dz)\) ruin classical AMG convergence rates!
- relatively small horizontal coupling terms, hard to smooth horizontal errors
  \(\Rightarrow\) Solvers (AMG and ILU) must take **aspect ratios** into account!

We developed a new **AMG solver** based on aggressive **semi-coarsening** (available in **ML/MueLu packages of Trilinos**)

See (Tezaur *et al.*, 2015), (Tuminaro *et al.*, 2016).
Greenland Controlled Weak Scalability Study

- Weak scaling study with fixed dataset, 4 mesh bisections.
- ~70-80K dofs/core.
- Conjugate Gradient (CG) iterative method for linear solves (faster convergence than GMRES).
- New AMG preconditioner developed by R. Tuminaro based on semi-coarsening (coarsening in $z$-direction only).
- Significant improvement in scalability with new AMG preconditioner over ILU preconditioner!
Greenland Controlled Weak Scalability Study

- Weak scaling study with fixed dataset, 4 mesh bisections.
- ~70-80K dofs/core.
- Conjugate Gradient (CG) iterative method for linear solves (faster convergence than GMRES).
- New AMG preconditioner developed by R. Tuminaro based on semi-coarsening (coarsening in $z$-direction only).
- Significant improvement in scalability with new AMG preconditioner over ILU preconditioner!

4 cores
334K dofs
8 km Greenland, 5 vertical layers

× $8^4$ scale up

16,384 cores
1.12B dofs(!)
0.5 km Greenland, 80 vertical layers

Significant improvement in scalability with new AMG preconditioner over ILU preconditioner!
Weak scalability: Antarctica

- **Weak scaling study**: 2.5M → 1.1B dofs, 16 → 8192 cores
- Initialized with realistic basal friction and temperature field from BEDMAP2.
- **Iterative linear solver**: GMRES.
- **Preconditioner**: ILU vs. new AMG based on aggressive semi-coarsening.

Antarctica is fundamentally different than Greenland: AIS contains large ice shelves (floating extensions of land ice).

- **ILU solver does not converge** for finest mesh resolution!

\[ A^{-1} \] will have large number of non-zeroes, so approximate inverse ILU preconditioner is ineffective.

Thin floating ice: ILU will not work well! Green’s function ~ constant in thin direction*

Thin grounded ice: ILU can work well w/ proper ordering

See (Tuminaro et al., *SISC*, 2016).

* For thin floating ice, the matrix inversion will have a large number of non-zero elements, making the approximate inverse ILU preconditioner ineffective.
Outline

1. Background
   • Motivation for climate & land-ice modeling
   • ISMs, ESMs & projects
   • Land-ice equations
   • Our codes: ALI, MALI

2. Algorithms and software
   • Discretization & meshes
   • Nonlinear solvers
   • Linear solvers
   • Performance-portability
   • Ice sheet initialization
   • Towards UQ

3. Simulations

4. Ongoing & future work
Performance-Portability via *Kokkos*

We need to be able to run *Albany Land-Ice* on **new architecture machines** (hybrid systems) and **manycore devices** (multi-core CPU, GPUs, Intel Xeon Phi, etc.).

**MPI** (inter-node parallelism) + **X** (intra-node parallelism)

- **Kokkos**: open-source C++ library that provides performance portability across diverse devices with different memory models.
  - A *programming model* as much as a software library.
  - Provides automatic access to OpenMP, CUDA, Pthreads, ...
  - Templated meta-programming: `parallel_for`, `parallel_reduce` (templated on an *execution space*).
  - Memory layout abstraction ("array of structs" vs. "struct of arrays", locality).

With *Kokkos*, you write an algorithm **once**, and just change a template parameter to get the optimal data layout for your hardware (e.g., `(i,j,k)` vs. `(k,i,j)`).

- **Finite element assembly** in *Albany Land-Ice* has been rewritten using *Kokkos* functors.
- Performance portability for **linear solvers** is an ongoing research topic within Trilinos.

*X* = OpenMP, CUDA, etc.  **https://github.com/kokkos/kokkos
**Kokkos-ification of Finite Element Assembly (FEA)**

```cpp
typedef Kokkos::OpenMP ExecutionSpace;
//typedef Kokkos::CUDA ExecutionSpace;
//typedef Kokkos::Serial ExecutionSpace;
template<typename ScalarT>
vectorGrad<ScalarT>::vectorGrad()
{
  Kokkos::View<ScalarT**, ExecutionSpace> vecGrad("vecGrad", numCells, numQP, numVec, numDim);
}
******************************************************************************
template<typename ScalarT>
void vectorGrad<ScalarT>::evaluateFields()
{
  Kokkos::parallel_for<ExecutionSpace> (numCells, *this);
}
******************************************************************************
template<typename ScalarT>
KOKKOS_INLINE_FUNCTION
void vectorGrad<ScalarT>::operator() (const int cell) const
{
  for (int cell = 0; cell < numCells; cell++)
    for (int qp = 0; qp < numQP; qp++)
      for (int dim = 0; dim < numVec; dim++)
        for (int i = 0; i < numDim; i++)
          vecGrad(cell, qp, dim, i) += val(cell, nd, dim) * basisGrad(nd, qp, i);
}
```

ExecutionSpace parameter tailors code for device (e.g., OpenMP, CUDA, etc.)
Cori (NERSC): 2,388 Haswell nodes [2 Haswell (32 cores)]
9,688 KNL nodes [1 Xeon Phi KNL (68 cores)]
Summit (OLCF): 4600 nodes [2 P9 (22 cores) + V100 (6 GPUs)]
Future targets: Aurora Intel GPU (ALCF), Frontier AMD GPU (OLCF)

Performance-portability of FEA in ALI has been tested across **multiple architectures**: Intel Sandy Bridge, Intel Skylake, IBM POWER8, IBM POWER9, Kepler/Pascal/Volta/Ampere GPUs, KNL Xeon Phi

**Antarctica performance monitoring**
Outline

1. Background
   • Motivation for climate & land-ice modeling
   • ISMs, ESMs & projects
   • Land-ice equations
   • Our codes: ALI, MALI

2. Algorithms and software
   • Discretization & meshes
   • Nonlinear solvers
   • Linear solvers
   • Performance-portability
   • Ice sheet initialization
   • Towards UQ

3. Simulations

4. Ongoing & future work
Inversion for Ice Sheet Initialization

**Goal:** find ice sheet initial state that:

- matches observations (e.g. surface velocity, temperature).
- matches present-day geometry (elevation, thickness).
- is in “equilibrium” with climate forcings (SMB).

**Available data/measurements:**

- Ice extent and surface topography.
- Surface velocity.
- Surface mass balance (SMB).
- Ice thickness $H$ (sparse measurements).

**Fields to be estimated:**

- Basal friction $\beta$, ice thickness $H$

**“Spin-up” approach:** initialize model with (imperfect/unknown) present state and integrate forward until states consistent with observations are reached.

- Can require **a lot of CPU time** (“spin-up time”): long timescale adjustments to past BC forcing requires a model “spin-up” of order $10^4$-$10^5$ years*.
- “Spun-up” initial conditions can result in **shocks**, which initiate large transients that can **derail** dynamic ice simulations*.

---

Deterministic Inversion

First-Order Stokes PDE-Constrained optimization problem for initial condition*:

\[
\text{minimize } \beta, H \quad m(u, H) \\
\text{s.t. FO Stokes PDEs}
\]

**Modeling Assumptions:** ice described by FO Stokes equations; ice close to mechanical equilibrium.

\[
m(u, H) = \int_{\Gamma} \frac{1}{\sigma_u^2} |u - u^{obs}|^2 ds \\
+ \int_{\Gamma} \frac{1}{\sigma_t^2} |\text{div}(UH) - \tau_s|^2 ds \\
+ \int_{\Gamma} \frac{1}{\sigma_H^2} |H - H^{obs}|^2 ds \\
+ \mathcal{R}(u, H)
\]

- \(U\): computed depth averaged velocity
- \(H\): ice thickness
- \(\beta\): basal sliding friction coefficient
- \(\tau_s\): surface mass balance (SMB)
- \(\mathcal{R}(u, H)\): regularization term
- \(\sigma\): standard deviation (weight of uncertainties)

Deterministic Inversion

First-Order Stokes PDE-Constrained optimization problem for initial condition*:

\[
\begin{align*}
\text{minimize} & \quad \beta, H \quad m(u, H) \\
\text{s.t.} & \quad \text{FO Stokes PDEs}
\end{align*}
\]

**Modeling Assumptions:** ice described by FO Stokes equations; ice close to mechanical equilibrium.

\[
m(u, H) = \int_\Gamma \frac{1}{\sigma_u^2} |u - u^{\text{obs}}|^2 ds + \int_\Gamma \frac{1}{\sigma_t^2} |\text{div}(UH) - \tau_s|^2 ds + \int_\Gamma \frac{1}{\sigma_H^2} |H - H^{\text{obs}}|^2 ds + R(u, H)
\]

- \(U\): computed depth averaged velocity
- \(H\): ice thickness
- \(\beta\): basal sliding friction coefficient
- \(\tau_s\): surface mass balance (SMB)
- \(R(u, H)\): regularization term
- \(\sigma\): standard deviation (weight of uncertainties)

Deterministic Inversion

**First-Order Stokes PDE-Constrained optimization problem for initial condition**: 

\[
\begin{align*}
\text{minimize} & \quad \beta, H \quad m(u, H) \\
\text{s.t.} & \quad \text{FO Stokes PDEs}
\end{align*}
\]

**Modeling Assumptions**: ice described by FO Stokes equations; ice close to mechanical equilibrium.

\[
m(u, H) = \int_{\Gamma} \frac{1}{\sigma_u^2} |u - u^{obs}|^2 ds + \int_{\Gamma} \frac{1}{\sigma_t^2} |\text{div}(UH) - \tau_s|^2 ds + \int_{\Gamma} \frac{1}{\sigma_H^2} |H - H^{obs}|^2 ds + R(u, H)
\]

- **surface velocity mismatch**
- **SMB mismatch**
- **thickness mismatch**
- **regularization terms**

\[
U: \text{computed depth averaged velocity} \\
H: \text{ice thickness} \\
\beta: \text{basal sliding friction coefficient} \\
\tau_s: \text{surface mass balance (SMB)} \\
R(u, H): \text{regularization term} \\
\sigma: \text{standard deviation (weight of uncertainties)}
\]

Solving FO Stokes PDE-constrained optimization problem for initial condition significantly reduces non-physical model transients!

### Deterministic Inversion Algorithm & Software

First-Order Stokes PDE-Constrained optimization problem for initial condition*:

\[
\text{minimize } \beta, H \quad m(u, H)
\]
\[
s.t. \text{ FO Stokes PDEs}
\]

Solved via embedded **adjoint-based PDE-constrained optimization** algorithm in Albany Land-Ice.

Approach efficiently computes **gradients** of \( m(u, H) \) by solving **linear adjoint PDEs**.

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Software</th>
</tr>
</thead>
<tbody>
<tr>
<td>Finite Element Method discretization</td>
<td>Albany</td>
</tr>
<tr>
<td>Quasi-Newton optimization (L-BFGS)</td>
<td>ROL</td>
</tr>
<tr>
<td>Nonlinear solver (Newton)</td>
<td>NOX</td>
</tr>
<tr>
<td>Krylov linear solvers</td>
<td>Belos+Ifpack2/Muelu</td>
</tr>
</tbody>
</table>

*Some details:*

- **Regularization**: Tikhonov.
- Total derivatives of objective functional \( m(u, H) \) computed using **adjoints** and **automatic differentiation** (Sacado package of Trilinos).
- **Gradient-based optimization**: limited memory BFGS initialized with Hessian of regularization terms (ROL) with backtrack linesearch.

Deterministic Inversion: 1km Greenland Initial Condition*

Deterministic Inversion: Common vs. Novel Approach*

SMB (m/yr) needed for equilibrium

| beta only | beta and H |

SMB (m/yr) from climate model (Ettema et al. 2009, RACMO2/GR)

target

Plot saturated. In many places field is ± hundreds m/yr.

High-Resolution Antarctica Optimal Initial Condition

Optimized surface speed for **variable-resolution Antarctic ice sheet initial condition**. Mesh resolution varies from ~40 km in slow moving EAIS interior to ~1.5 km in regions with ice shelves, ice streams, and below-sea level bedrock elevation.
Velocity-Temperature Coupling

• MALI default coupling between FO Stokes and temperature is **sequential**

• We are working towards **fully-coupled flow + temperature** model
  - Enables computation of **self-consistent** ice sheet initial state (with ice temperature).

• Current implementation in Albany Land-Ice: steady-state **enthalpy equation** coupled monolithically with **FO Stokes equations**
  
  **Enthalpy equation:** \( \mathbf{u} \cdot \nabla h + \nabla \cdot \mathbf{q} = \tau : \dot{\varepsilon} \)

  \( h = \) enthalpy  
  \( \tau = \) dissipation heat  
  \( \mathbf{q} = \) total heat flux

• Challenges include **strong nonlinearity** of basal BC due to **phase changes** and **robust solvers**.

**Strategy:** approximate enthalpy/melting graph at bed by smooth function, perform parameter **continuation** to smoothly transition from cold to temperate ice (left).

**Developing robust linear solvers** for coupled velocity-temperature equations is WIP.
Simultaneous Velocity-Temperature Initialization (Inversion)

First-Order Stokes PDE-Constrained optimization problem for initial condition:

\[
\begin{align*}
\text{minimize } & \beta, H \quad m(u, H) \\
\text{s.t. } & \text{FO Stokes PDEs + Enthalpy PDE}
\end{align*}
\]

- With an implicit steady-state coupled temperature-velocity model, one can obtain **self-consistent** state in **one shot**.
- Initialization capability is **unmatched** by other land-ice codes:
  - Typically ~10K years are needed to equilibrate ice temperature
  - Our solver **robustly** computes the steady-state temperature coupled w/ velocity at every iteration of the optimization

*Left*: Computed basal temperature

*Right*: Thawed/frozen map from MacGregor *et al.*, JGR, 2016

modeled ice temperature
Outline

1. Background
   • Motivation for climate & land-ice modeling
   • ISMs, ESMs & projects
   • Land-ice equations
   • Our codes: ALI, MALI

2. Algorithms and software
   • Discretization & meshes
   • Nonlinear solvers
   • Linear solvers
   • Performance-portability
   • Ice sheet initialization
   • Towards UQ

3. Simulations

4. Ongoing & future work
Uncertainty Quantification*

**Goal:** obtain PDF of initial condition using Bayesian inference and propagate this PDF through model to get PDF of *total ice mass loss/gain during 21st century*.

**Stage 1:** Estimate ice sheet initial condition (MAP point).

**Stage 2:** Update prior uncertainty in ice sheet initial condition using observational data and steady state model.

**Stage 3:** Propagate uncertain initial condition through ice-sheet evolution model.

* Jakeman et al. (in prep), 2021.
Uncertainty Quantification*

**Goal:** obtain PDF of initial condition using Bayesian inference and propagate this PDF through model to get PDF of total ice mass loss/gain during 21st century

**Stage 1:** Estimate ice sheet initial condition (MAP point).

**Stage 2:** Update prior uncertainty in ice sheet initial condition using observational data and steady state model.

**Stage 3:** Propagate uncertain initial condition through ice-sheet evolution model.

Very challenging! Lots of obstacles, e.g., curse of dimensionality.

* Jakeman et al. (in prep), 2021.
Outline

1. Background
   • Motivation for climate & land-ice modeling
   • ISMs, ESMs & projects
   • Land-ice equations
   • Our codes: ALI, MALI

2. Algorithms and software
   • Discretization & meshes
   • Nonlinear solvers
   • Linear solvers
   • Performance-portability
   • Ice sheet initialization
   • Towards UQ

3. Simulations

4. Ongoing & future work
Model Validation

Our model has been validated* using data from two satellites: ICESat, GRACE.

Surface elevation predictions (states) agree pretty well with GLAS (Geoscience Laser Altimeter System) aboard ICESat: mean differences are <1 m

|------------------|-------------|

Forcings**:

- **SMB-only**: Mass change computed by solving an ISM forced w/ RACMO SMB (2003-2012)
- **SMB+FF**: Mass change computed as in SMB-only with additional flux term on significant ice streams
- **RACMO**: mass change computed directly from SMB without using an ice sheet model

ABUMIP*-Antarctica Experiment

**Basic idea**: instantaneously remove all ice shelves and see what happens in the next 200 years, preventing any floating ice from ever forming again → Provides an *extreme upper bound* on SLR contributions from Antarctica

~32M unknowns solved for on 6400 procs, with average model throughput of ~120 simulated yrs/wall clock day.

*Movie Above*: 200 year MALI Antarctic ice sheet simulation after instantaneous removal of all floating ice shelves

* Courtesy of M. Hoffman, S. Price, T. Zhang, N. Woods, J. Patchet (LANL)
**ABUMIP*-Antarctica Experiment**

**Basic idea**: instantaneously *remove all ice shelves* and see what happens in the next 200 years, preventing any floating ice from ever forming again → Provides an *extreme upper bound* on SLR contributions from Antarctica.

---

**Figure Above**: Antarctic ice sheet simulation after instantaneous removal of all floating ice shelves at year 200

*~32M unknowns solved for on 6400 procs*, with average model throughput of *~120 simulated yrs/wall clock day.*

*Courtesy of M. Hoffman, S. Price, T. Zhang, N. Woods, J. Patchet (LANL)*
LARMIP*-Antarctica Experiment

Similar to control run (forced with historical observations) in most parts of Antarctica, but includes **warmer ocean water** flowing into the cavity beneath the **Filchner-Ronne Ice Shelf** → provides example of ice sheet’s response to **aggressive melting and thinning**

https://www.youtube.com/watch?v=Wt0TvNjYsOs&feature=youtu.be

* Linear Antarctic Response Model Intercomparison Project
Simulations: Ice Sheets & SLR under ISMIP6*

Future Antarctica sea level contribution under rapid ice shelf melting (top left) and ice shelf collapse (bottom right).

PDF of future sea-level rise from all land ice from emulation of ISMIP6 and GlacierMIP model projections (upper right).

~20% of ice sheet contributions from DOE-developed models

Most other models are 2D, ad hoc hybrids, or are run at relatively coarse resolution
MALI Thwaites Glacier Simulation

- Movie shows *Thwaites Glacier* retreat simulation under parameterized submarine melting.
- 250 year *regional simulation* with “present day” initial condition.
- Investigate importance of *CDW* depth changes due to climate variability.
- When *climate variability* in sub-shelf forcing is accounted for, we get a *distribution* of possible SLR curves.

* CDW = Circumpolar Deep Water.
MALI & E3SM Coupling

- **Global, coupled** E3SM simulation with sub-ice shelf circulation + pre-industrial forcing + static ice shelves *(illustration/spin-up over ~7 yrs).*
- RRS30to10km mesh (eddy permitting).

MALI is (partially) coupled to E3SM and currently supports **static ice shelves** and **fixed grounding lines** (enabling dynamic ice shelves is WIP).

**Top:** sea-surface salinity

**Right:** ocean bottom temperature
Outline

1. Background
   • Motivation for climate & land-ice modeling
   • ISMs, ESMs & projects
   • Land-ice equations
   • Our codes: ALI, MALI

2. Algorithms and software
   • Discretization & meshes
   • Nonlinear solvers
   • Linear solvers
   • Performance-portability
   • Ice sheet initialization
   • Towards UQ

3. Simulations

4. Ongoing & future work
Probabilistic Sea-Level Projections from Ice Sheet and Earth System Models (ProSPect) is a new 5 year (2017-2022) SciDAC project on:

1) Ice sheet and ocean model **physics** critical for accurate projections of sea-level change (e.g., subglacial hydrology, damage evolution + fracture + calving)
2) Ice sheet, ocean, and ESM **coupling** critical for accurate projections of sea-level change
3) Ice sheet model **initialization** and **optimization** methods needed for realistic coupling of ISMs and ESMs
4) Frameworks for quantifying parametric and structural ice sheet model **uncertainties**
5) **Performance portability** on new, heterogeneous HPC architectures

New developments will be targeted at **standalone** and **coupled** simulations of sea-level rise from ice sheets
Summary

- **Actionable projections of climate change** and **sea-level rise impacts** are important worldwide!

- A **mature ice-sheet modeling capability** (high-fidelity, high-performance) was developed as a part of the PISCEES & ProSPect SciDAC projects. This talk described the following aspects of creating this capability:
  
  - **Equations, algorithms, software** used in ice sheet modeling.
  - The development of a finite element land ice solver known as **Albany Land-Ice** written using the libraries of the **Trilinos** libraries.
  - **Coupling** of Albany Land-Ice to **MPAS LI** codes for transient simulations of ice sheet evolution.
  - Some **advanced concepts** in ice sheet modeling: ice sheet initialization/inversion.

- Related capabilities on the E3SM side are rapidly **maturing**.

- Ongoing projects are focusing on the remaining work (physics, coupling, uncertainty quantification frameworks) necessary to provide **SLR projections and uncertainties**.
References


References (cont’d)


https://www.sandia.gov/~ikalash/journal.html
Sandia Land-Ice Work In-The-News!

Ice sheet modeling of Greenland, Antarctica helps predict sea-level rise

Michael Padilla

The Greenland and Antarctic ice sheets will make a dominant contribution to 21st century sea-level rise if current climate trends continue. However, predicting the expected loss of ice sheet mass is difficult due to the complexity of modeling ice sheet behavior.

Computing (SciDAC) program. PISEES is a multi-lab, multi-university endeavor that includes researchers from Sandia, Los Alamos, Lawrence Berkeley, and Oak Ridge national laboratories; the Massachusetts Institute of Technology; Florida State University; the University of Bristol; the University of Texas Austin; the University of South Carolina; and New York University.

Sandia’s biggest contribution to PISEES has been an analysis tool: a land-ice solver called Albany/FELIX (Finite Elements for Land ice Experiments). The tool is based on equations that simulate ice flow over the Greenland and Antarctic ice sheets and is being coupled to Earth models through the Accelerated Climate for Energy (ACME) project.

“One of the goals of PISEES is to create a land-ice solver that is scalable, fast, and robust on computational scales,” says computational scientist Irina Tezaur, a lead developer of Albany/FELIX. Not only did the new solver need to be reliable and efficient, but it was critical that the team develop a solver capable of running on new and emerging computers. And so, with advanced

https://www.sandia.gov/~ikalash
Students: please consider Sandia and other national labs as a potential employer for summer internships and when you graduate!

- Sandia is a **multidisciplinary** national lab and Federally Funded Research & Development Center (FFRDC).
- Contractor for U.S. DOE’s National Nuclear Security Administration (**NNSA**).
- **Two main sites:** Albuquerque, NM and Livermore, CA.
Careers at Sandia

**Students:** please consider Sandia and other national labs as a potential employer for summer internships and when you graduate!

- **Sandia is a great place to work!**
  - Lots of **interesting** problems that require **fundamental research** in applied math/computational science and impact **mission-critical applications**.
  - Great **work/life balance**.

- **Opportunities** at/with Sandia:
  - Interns (including PSAAP)
  - Post docs
  - Several prestigious post doctoral fellowships (von Neumann, Truman, Hruby)
  - Staff

*Please see: [www.sandia.gov/careers](http://www.sandia.gov/careers) for info about current opportunities.*
Backup Slides
Motivation

Department of Energy (DOE) interests in climate change and sea-level rise:

• “Addressing the effects of climate change is a top priority of the DOE.”*

• DOE report on energy sector vulnerabilities: “… higher risks to energy infrastructure located along the coasts thanks to sea level rise, the increasing intensity of storms, and higher storm surge and flooding.”**

*http://energy.gov/science-innovation/climate-change
**http://energy.gov/articles/climate-change-effects-our-energy
A Hierarchy of Ice Sheet Models

Full Stokes Flow Model
continental or regional simulations

Higher-Order Models
e.g. First Order Stokes/Blatter-Pattyn Model
continental or regional simulations

Hybrid Models
e.g. SIA+SSA, SIA+FS, SS+FS
regional simulations of ice sheet/shelf/stream

Zero-th Order Models
Shallow Ice Approximation (SIA)
Shallow Shelf Approximation (SSA)
regional of ice streams or shelves

# A Hierarchy of Ice Sheet Models (ISMs)

<table>
<thead>
<tr>
<th>Model Name</th>
<th>Terms Kept</th>
<th>Comments</th>
<th>Validity</th>
</tr>
</thead>
<tbody>
<tr>
<td>Stokes</td>
<td>All</td>
<td>3D model for ((u, p))</td>
<td>continental scale</td>
</tr>
<tr>
<td>First-Order Stokes/Blatter-Pattyn(^1)</td>
<td>(O(\delta))</td>
<td>3D model for ((u_1, u_2))</td>
<td>continental scale</td>
</tr>
<tr>
<td>L1L1, L1L2(^2)</td>
<td>(O(\delta))</td>
<td>Depth integrated, 2D models for ((u_1, u_2))</td>
<td>Antarctica</td>
</tr>
<tr>
<td>Shallow Ice (SIA)(^3)</td>
<td>(O(1))</td>
<td>Depth integrated, 2D model for ((u_1, u_2))</td>
<td>grounded ice with frozen bed</td>
</tr>
<tr>
<td>Shallow Shelf (SSA)(^4)</td>
<td>(O(1))</td>
<td>Closed form for (u_1)</td>
<td>shelves or fast sliding grounded ice</td>
</tr>
</tbody>
</table>

## A Hierarchy of Ice Sheet Models (ISMs)

<table>
<thead>
<tr>
<th>Model Name</th>
<th>Terms Kept</th>
<th>Comments</th>
<th>Validity</th>
</tr>
</thead>
<tbody>
<tr>
<td>Stokes</td>
<td>All</td>
<td>3D model for ((u, p))</td>
<td>continental scale</td>
</tr>
<tr>
<td>First-Order Stokes/Blatter-Pattyn(^1)</td>
<td>(O(\delta))</td>
<td>3D model for ((u_1, u_2))</td>
<td>continental scale</td>
</tr>
<tr>
<td>L1L1, L1L2(^2)</td>
<td>(O(\delta))</td>
<td>Depth integrated, 2D models for ((u_1, u_2))</td>
<td>Antarctica</td>
</tr>
<tr>
<td>Shallow Ice (SIA)(^3)</td>
<td>(O(1))</td>
<td>Depth integrated, 2D model for ((u_1, u_2))</td>
<td>grounded ice with frozen bed</td>
</tr>
<tr>
<td>Shallow Shelf (SSA)(^4)</td>
<td>(O(1))</td>
<td>Closed form for (u_1)</td>
<td>shelves or fast sliding grounded ice</td>
</tr>
</tbody>
</table>

- Stokes flow model is “**gold standard**” but expensive.

---

A Hierarchy of Ice Sheet Models (ISMs)

<table>
<thead>
<tr>
<th>Model Name</th>
<th>Terms Kept</th>
<th>Comments</th>
<th>Validity</th>
</tr>
</thead>
<tbody>
<tr>
<td>Stokes</td>
<td>All</td>
<td>3D model for ( (u, p) )</td>
<td>continental scale</td>
</tr>
<tr>
<td>First-Order Stokes/Blatter-Pattyn(^1)</td>
<td>( O(\delta) )</td>
<td>3D model for ( (u_1, u_2) )</td>
<td>continental scale</td>
</tr>
<tr>
<td>L1L1, L1L2(^2)</td>
<td>( O(\delta) )</td>
<td>Depth integrated, 2D models for ( (u_1, u_2) )</td>
<td>Antarctica</td>
</tr>
<tr>
<td>Shallow Ice (SIA)(^3)</td>
<td>( O(1) )</td>
<td>Depth integrated, 2D model for ( (u_1, u_2) )</td>
<td>grounded ice with frozen bed</td>
</tr>
<tr>
<td>Shallow Shelf (SSA)(^4)</td>
<td>( O(1) )</td>
<td>Closed form for ( u_1 )</td>
<td>shelves or fast sliding grounded ice</td>
</tr>
</tbody>
</table>

- Stokes flow model is "gold standard" but expensive.
- **Simplified models** are derived from full Stokes model and take advantage of the fact that ice sheets are thin: \( \delta \ll 1 \).
Shallow Shelf and Shallow Ice Approximation

**Ice regime:**
- Shelves or fast sliding grounded ice
- Ice regime: grounded ice with frozen bed

**Shallow Ice Approximation**

\[
\epsilon(u) = \begin{pmatrix}
0 & 0 & 0.5u_z \\
0 & 0 & 0.5v_z \\
0 & 0 & w_z
\end{pmatrix}
\]

\[p = \rho g (s - z)\]

**FO Stokes** \((u, v) \) in \(\Omega \in \mathbb{R}^3\)

\[
\epsilon(u) = \begin{pmatrix}
u_x & 0.5(u_y + v_x) & 0 \\
0.5(u_y + v_x) & v_y & 0 \\
0 & 0 & w_z
\end{pmatrix}
\]

\[p = \rho g (s - z) - 2\mu(u_x + v_y)\]

**Shallow Shelf Approximation**

\(\text{SIA}(u, v) \) in \(\Omega \in \mathbb{R}^3\)

\(\text{SSA}(u, v) \) in \(\Sigma \in \mathbb{R}^2\)

**Discussion:**
- **Neither SIA nor SSA applies at continental scale.**
- SIA and SSA are referred to as “zero-th order” models
- Both models have two unknowns \((u, v)\).
- SSA is 2D model obtained by vertically integrating the equations.
ISM Computation Cost in ESM

High-res climate model processor layout

<table>
<thead>
<tr>
<th>grid size</th>
<th>component</th>
<th>horizontal</th>
<th>vertical</th>
</tr>
</thead>
<tbody>
<tr>
<td>25km</td>
<td>ATM/LND</td>
<td>0.8M</td>
<td>72</td>
</tr>
<tr>
<td>18-6km</td>
<td>OCN/ICE</td>
<td>3.7M</td>
<td>80</td>
</tr>
<tr>
<td>2-20km</td>
<td>AIS ISM</td>
<td>1.6M</td>
<td>10</td>
</tr>
</tbody>
</table>

- **ISM throughput**: 1 SYPD
  (simulated year per wallclock day)

- **ISM cost**: 4M core-hours per simulated year
Numerical & Computational Challenges

- **Mesh adaptivity** close to the grounding line.
- FO Stokes equations are **highly nonlinear**.
- Large, **thin geometries** (thickness up to 4km, horizontal extension 1000s of kms).
  - Gives rise to meshes with **bad aspect ratios** and **poorly conditioned** linear systems.
- **Boundary conditions** pose challenges to solvers.
- **Porting** of software to **new architectures** (hybrid systems, GPUs, etc.).
- **Initialization/estimation of unknown parameters** (basal friction, thickness, etc.).
- **Uncertainty quantification**.
  - Curse of dimensionality!
- **Thickness evolution** (ice advancement/retreat)
  - Sequential coupling with FO Stokes equations gives rise to very small time-steps by CFL condition!
- **Phase changes** (temperature equation).
- **Coupling to climate components**.
Mesh Adaptivity

**PAALS = Parallel Albany Adaptive Loop with SCOREC**

- In collaboration with **Rensselaer Polytechnical Institute** (M. Shephard, C. Smith, B. Granzow): added mesh adaptation capabilities (PAALS) to *Albany*.

**PAALS provides:**

- Fully-coupled, *in-memory adaptation* and solution transfer services.
- *Parallel mesh infrastructure* and services via *PUMI* (Parallel Unstructured Mesh Infrastructure): an efficient, distributed mesh data structure that supports adaptivity.
- Predictive *dynamic load balancing* via *ParMetis/Zoltan + ParMA*.
- SPR**-based generalized *error estimation* of *velocity gradient* drives adaptation.
- *Performance portability* to GPUs via *Kokkos*.

**Ryder glacier (north coast)**

*Left*: before mesh adaptation; *Right*: after mesh adaptation

**Super-convergent Patch Recovery**: technique for estimating $\nabla u$ using quadratic approximation within a patch of elements.

*SCOREC = Scientific Computation Research Center at RPI: https://github.com/SCOREC*
Mesh Convergence Studies

**Stage 1:** solution verification on 2D MMS problems we derived.

**Stage 2:** code-to-code comparisons on canonical ice sheet problems.

**Stage 3:** full 3D mesh convergence study on Greenland w.r.t. reference solution.

Are the Greenland problems resolved? Is theoretical convergence rate achieved?
Mesh Partitioning & Vertical Refinement

Mesh convergence studies led to some useful practical recommendations (for ice sheet modelers and geo-scientists)!

- **Partitioning matters**: good solver performance obtained with 2D partition of mesh (all elements with same $x$, $y$ coordinates on same processor - right).

- **Number of vertical layers matters**: more gained in refining # vertical layers than horizontal resolution (below – relative errors for Greenland).

<table>
<thead>
<tr>
<th>Horiz. res./vert. layers</th>
<th>5</th>
<th>10</th>
<th>20</th>
<th>40</th>
<th>80</th>
</tr>
</thead>
<tbody>
<tr>
<td>8km</td>
<td>2.0e-1</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4km</td>
<td>9.0e-2</td>
<td>7.8e-2</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2km</td>
<td>4.6e-2</td>
<td>2.4e-2</td>
<td>2.3e-2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1km</td>
<td>3.8e-2</td>
<td>8.9e-3</td>
<td>5.5e-3</td>
<td>5.1e-3</td>
<td></td>
</tr>
<tr>
<td>500m</td>
<td>3.7e-2</td>
<td>6.7e-3</td>
<td>1.7e-3</td>
<td>3.9e-4</td>
<td>8.1e-5</td>
</tr>
</tbody>
</table>

Vertical refinement to 20 layers recommended for 1km resolution over horizontal refinement.
Importance of Node Ordering & Mesh Partitioning

Our studies revealed that node ordering and mesh partitioning matters for linear solver performance, especially for the ILU preconditioner!

• It is essential that incomplete factorization accurately captures vertical coupling, which is dominant due to anisotropic mesh.

• This is accomplished by:
  • Ensuring all points along a vertically extruded grid line reside within a single processor (“2D mesh partitioning”; top right).
  • Ordering the equations such that grid layer $k$’s nodes are ordered before all dofs associated with grid layer $k + 1$ (“row-wise ordering”; bottom right).
**Improved Linear Solver Performance through Hinge Removal**

- Islands and certain hinged peninsulas lead to **solver failures**

  - We have developed an algorithm to detect/remove problematic **hinged peninsulas & islands** based on coloring and repeated use of connected component algorithms (Tuminaro *et al.*, 2016).
  - Solves are ~2x faster with hinges removed.
  - Current implementation is MATLAB, but working on C++ implementation for integration into dycores.

<table>
<thead>
<tr>
<th>Resolution</th>
<th>ILU – hinges</th>
<th>ILU – no hinges</th>
<th>ML – hinges</th>
<th>ML – no hinges</th>
</tr>
</thead>
<tbody>
<tr>
<td>8km/5 layers</td>
<td>878 sec, 84 iter/solve</td>
<td>693 sec, 71 iter/solve</td>
<td>254 sec, 11 iter/solve</td>
<td>220 sec, 9 iter/solve</td>
</tr>
<tr>
<td>4km/10 layers</td>
<td>1953 sec, 160 iter/solve</td>
<td>1969 sec, 160 iter/solve</td>
<td>285 sec, 13 iter/solve</td>
<td>245 sec, 12 iter/solve</td>
</tr>
<tr>
<td>2km/20 layers</td>
<td>10942 sec, 710 iter/solve</td>
<td>5576 sec, 426 iter/solve</td>
<td>482 sec, 24 iter/solve</td>
<td>294 sec, 15 iter/solve</td>
</tr>
<tr>
<td>1km/40 layers</td>
<td>--</td>
<td>15716 sec, 881 iter/solve</td>
<td>668 sec, 34 iter/solve</td>
<td>378 sec, 20 iter/solve</td>
</tr>
</tbody>
</table>

**Greenland Problem**
Spherical Grids

- Current ice sheet models are derived using planar geometries – reasonable, especially for Greenland.
- The effect of Earth’s curvature is largely unknown – may be nontrivial for Antarctica.
- We have derived a FO Stokes model on sphere using stereographic projection.
Deterministic Inversion: Stiffening Factor

Glen’s viscosity with **stiffening/damage**: 

\[ \mu^*(x, y, z) = \phi(x, y)\mu(x, y, z) \]

where \( \phi(x, y) = \) stiffening/damage factor that accounts for modeling errors in rheology.

AIS inversion for \( \beta(x, y) \) and \( \phi(x, y) \) simultaneously.
**UQ Problem Definition**

**QoI in Ice Sheet Modeling:** total ice mass loss/gain during 21st century → *sea level change prediction.*

**Sources of uncertainty affecting this QoI include:**
- Climate forcings (e.g., surface mass balance).
- Basal friction ($\beta$).
- Ice sheet thickness ($h$).
- Geothermal heat flux.
- Model parameters (e.g., Glen’s flow law exponent).

**UQ Workflow**

*Stage 1:*
Estimate ice sheet initial condition (MAP point).

*Stage 2:*
Update prior uncertainty in ice sheet initial condition using observational data and steady state model.

*Stage 3:*
Propagate uncertain initial condition through ice-sheet evolution model.

**Deterministic inversion**

**Bayesian calibration**

**Forward propagation**

**Basal sliding BC:**

$$2\mu \dot{e}_i \cdot n + \beta u_i = 0,$$ on $\Gamma_\beta$

$$\mu = \frac{1}{2} A^{-\frac{1}{n}} \left( \frac{1}{2} \sum_{ij} \dot{e}_{ij}^2 + \gamma \right)^{\frac{1}{2n-2}}$$

$n = $ Glen’s law exponent
Bayesian Inference

UQ Workflow

Stage 1:
Estimate ice sheet initial condition (MAP point).

Stage 2:
Update prior uncertainty in ice sheet initial condition using observational data and steady state model.

Stage 3:
Propagate uncertain initial condition through ice-sheet evolution model.

Goal: solve inverse problem for ice sheet initial state but in Bayesian framework.

- Naïve parameterization: represent each degree of freedom on mesh be an uncertain variable

\[ \beta(x) = (z_1, z_2, \ldots, z_{n_{dof}}) \]

Intractable due to curse of dimensionality: \( n_{dof} = O(100K) \! \)

- To circumvent this difficulty: assume \( \beta(x) \) can be represented in reduced basis (e.g., KLE modes, Hessian eigenvectors*) centered around mean \( \bar{\beta}(x) \):

\[ \log(\beta(x)) = \log(\bar{\beta}) + \sum_{i=1}^{d} \sqrt{\lambda_i} \phi_i(x)z_i \]

- Mean field \( \bar{\beta}(x) = \) initial condition.

Deterministic inversion is consistent with Bayesian analog: it is used to find the MAP point of posterior.
Bayesian Inference Assumptions

- **Likelihood** is: $\hat{\pi}_{\text{hood}}(z) = e^{-m_{\text{lin}}(z)}$

- **Normal Laplace posterior** given by:
  \[ \pi_{\text{pos}}(z) = C_{\text{evid}}^{-1} \hat{\pi}_{\text{hood}}(z) \pi_{\text{pr}}(z) \]
  where $C_{\text{evid}} = \int \hat{\pi}_{\text{hood}}(z) \pi_{\text{pr}}(z) dz$.

- **Additive Gaussian noise model**: $y^{\text{obs}} = f(z) + \epsilon$, $\epsilon \sim N(0, \Gamma_{\text{obs}})$

  \[ m(z) = \frac{1}{2} (y^{\text{obs}} - f(z))^T \Gamma_{\text{obs}}^{-1} (y^{\text{obs}} - f(z)) \]

  ⇒ Mismatch functional to be minimized:

  \[ m(z) = \frac{1}{2} (y^{\text{obs}} - f(z))^T \Gamma_{\text{obs}}^{-1} (y^{\text{obs}} - f(z)) \]

- **Gaussian prior** with exponential covariance and mean $z_{\text{MAP}} = \bar{\beta}$.

**Evaluation of misfit Hessian is expensive!**
⇒ further approximation required.

Notation*:
- $y^{\text{obs}}$: observations
- $z$: random params
- $f(z)$: deterministic map from params to observables.

Bayesian Inference Workflow

1. Dimension reduction via KLE
2. Dimension reduction via AS
3. Quadratic PCE over active variables
4. Laplace posterior at MAP *

Two-part **dimension reduction** procedure to obtain modes \( \phi_i(x) \)

Procedure for computing **covariance of normal Laplace posterior**, \( \Gamma_{\text{post}} \)

**Abbreviations**

- **KLE** = Karhunen-Loeve Expansion
- **AS** = Active Subspace
- **PCE** = Polynomial Chaos Expansion
- **MAP** = Maximum a Posteriori

* Bui-Thanh, Ghattas, Martin, Stadler, SISC, 2013.
GIS Bayesian Inference via KLE + AS

- **Above:** marginal distributions of Gaussian posterior computed using KLE vs. KLE+AS; *any shift from mean of 0 is due to observations.*
  - KLE eigenvectors have variance and mean close to prior.
  - Data-informed eigenvectors have smaller variance and are most shifted w.r.t. prior distribution (as expected).

\[ C(r_1, r_2) = \exp \left( - \frac{(r_1 - r_2)^2}{L^2} \right) \]

* Value of \( d \) was obtained via cross-validation.
Bayesian Inference

• There are many sources of uncertainty, e.g.
  ➢ Climate forcing (e.g., surface mass balance)
  ➢ Basal friction
  ➢ Bedrock topography (noisy and sparse data)
  ➢ Geothermal heat flux
  ➢ Modeling errors
  ➢ Model parameters (e.g., Glen's Flow Law exponent)
Bayesian Inference

- There are many sources of uncertainty, e.g.
  - Climate forcing (e.g., surface mass balance)
  - Basal friction
  - Bedrock topography (noisy and sparse data)
  - Geothermal heat flux
  - Modeling errors
  - Model parameters (e.g., Glen's Flow Law exponent)

We focus initially only in uncertainty in **basal friction** $\beta$. 
Bayesian Inference

- There are many sources of uncertainty, e.g.
  - Climate forcing (e.g., surface mass balance)
  - Basal friction
  - Bedrock topography (noisy and sparse data)
  - Geothermal heat flux
  - Modeling errors
  - Model parameters (e.g., Glen's Flow Law exponent)

- **Bayes’ Theorem:** assume prior distribution, update using data:

\[
\pi(\theta | d) = \frac{\text{likelihood} \cdot \text{prior}}{\pi(d)} = \frac{\pi(d | \theta) \cdot \pi(\theta)}{\int \pi(d | \theta) \cdot \pi(\theta) \, d\theta}
\]
Bayesian Inference

- There are many sources of uncertainty, e.g.
  - Climate forcing (e.g., surface mass balance)
  - Basal friction
  - Bedrock topography (noisy and sparse data)
  - Geothermal heat flux
  - Modeling errors
  - Model parameters (e.g., Glen's Flow Law exponent)

- **Bayes’ Theorem**: assume prior distribution; update using data:

  \[
  \pi(\theta | d) \quad \text{posterior} = \frac{\pi(d | \theta) \pi(\theta)}{\pi(d)} = \frac{\int \pi(d | \theta) \pi(\theta) d\theta}{\pi(d)}
  \]

- **Naïve parameterization**: represent each degree of freedom on mesh be an uncertain variable \(\beta(x) = (z_1, z_2, ..., z_{n_{dof}})\)

  Intractable due to **curse of dimensionality**: \(n_{dof} = O(100K)\)!
Bayesian Inference

- There are many sources of uncertainty, e.g.
  - Climate forcing (e.g., surface mass balance)
  - Basal friction
  - Bedrock topography (noisy and sparse data)
  - Geothermal heat flux
  - Modeling errors
  - Model parameters (e.g., Glen's Flow Law exponent)

- **Bayes' Theorem**: assume prior distribution; update using data:

  \[
  \pi(\theta | d) = \frac{\underbrace{\pi(d | \theta) \pi(\theta)}_{\text{likelihood prior}}}{\pi(d)} = \frac{\int \pi(d | \theta) \pi(\theta) \, d\theta}{\pi(d)}
  \]

- **Naïve parameterization**: represent each degree of freedom on mesh be an uncertain variable \( \beta(x) = (z_1, z_2, ..., z_{n_{dof}}) \)

We focus initially only in uncertainty in **basal friction** \( \beta \).

Intractable due to **curse of dimensionality**: \( n_{dof} = O(100K)! \)
**Bayesian Inference**

**Approach 1: KLE + PCE + MCMC**

- **KLE = Karhunen Loeve Expansion:** assume $\beta(x)$ can be represented in *reduced basis* of KLE modes centered around mean $\bar{\beta}(x)$:

\[
\log(\beta(x)) = \log(\bar{\beta}) + \sum_{i=1}^{d} \sqrt{\lambda_i} \phi_i(x) z_i
\]

- **PCE = Polynomial Chaos Expansion:** create PCE emulator for mismatch (over surface velocity, SMB, thickness) discrepancy.

- **MCMC = Markov Chain Monte Carlo:** do MCMC calibration using PCE emulator to infer Maximum A Posteriori (MAP) point.
Bayesian Inference

**Approach 1: KLE + PCE + MCMC**

- **KLE = Karhunen Loeve Expansion**: assume $\beta(x)$ can be represented in **reduced basis** of KLE modes centered around mean $\bar{\beta}(x)$:

  $$\log(\beta(x)) = \log(\bar{\beta}) + \sum_{i=1}^{d} \sqrt{\lambda_i} \phi_i(x) z_i$$

- **PCE = Polynomial Chaos Expansion**: create PCE emulator for mismatch (over surface velocity, SMB, thickness) discrepancy.

- **MCMC = Markov Chain Monte Carlo**: do MCMC calibration using PCE emulator to infer Maximum A Posteriori (MAP) point.

**Upshots:**

😊 Can obtain **arbitrary** posterior distribution.
Bayesian Inference

**Approach 1:** KLE + PCE + MCMC

- **KLE = Karhunen Loeve Expansion:** assume $\beta(x)$ can be represented in *reduced basis* of KLE modes centered around mean $\bar{\beta}(x)$:

  $$\log(\beta(x)) = \log(\bar{\beta}) + \sum_{i=1}^{d} \sqrt{\lambda_i} \phi_i(x) z_i$$

- **PCE = Polynomial Chaos Expansion:** create PCE emulator for mismatch (over surface velocity, SMB, thickness) discrepancy.

- **MCMC = Markov Chain Monte Carlo:** do MCMC calibration using PCE emulator to infer Maximum A Posteriori (MAP) point.

**Upshots:**

- Can obtain arbitrary posterior distribution.

**Issues:**

- KLE requires *correlation length parameter*, which is unknown.
- MCMC only lets you use $O(10)$ KLE modes – many more are needed to represent basal friction field $O(1000)$; more modes needed for finer resolution problems.
**Bayesian Inference**

**Approach 1: KLE + PCE + MCMC**

- **KLE = Karhunen Loeve Expansion:** assume $\beta(x)$ can be represented in *reduced basis* of KLE modes centered around mean $\bar{\beta}(x)$:

  $$\log(\beta(x)) = \log(\bar{\beta}) + \sum_{i=1}^{d} \sqrt{\lambda_i} \phi_i(x) z_i$$

- **PCE = Polynomial Chaos Expansion:** create PCE emulator for mismatch (over surface velocity, SMB, thickness) discrepancy.

- **MCMC = Markov Chain Monte Carlo:** do MCMC calibration using PCE emulator to infer Maximum A Posteriori (MAP) point.

**Upshots:**

😊 Can obtain *arbitrary* posterior distribution.

**Issues:**

😊 KLE requires *correlation length parameter*, which is unknown.

😊 MCMC only lets you use $O(10)$ KLE modes – many more are needed to represent basal friction field $O(1000)$; more modes needed for finer resolution problems.
Bayesian Inference

**Approach 2:** Normal Approximation + Low Rank Laplace Approximation*

- Gaussian prior, likelihood $\Rightarrow$ **Gaussian posterior:**
  \[
  \pi_{\text{pos}}(z \mid y^{\text{obs}}) = N(z_{\text{MAP}}, \Gamma_{\text{post}})
  \]

Bayesian Inference

**Approach 2:** Normal Approximation + Low Rank Laplace Approximation*

- Gaussian prior, likelihood ⇒ *Gaussian posterior:* \[ \pi_{\text{pos}}(z \mid y^{\text{obs}}) = N(z_{\text{MAP}}, \Gamma_{\text{post}}) \]

- **Linearize** parameter-to-observable map around MAP point:
  \[ y^{\text{obs}} = f(z) + \epsilon \approx f(z_{\text{MAP}}) + F(z - z_{\text{MAP}}) + \epsilon \]

Bayesian Inference

**Approach 2:** Normal Approximation + Low Rank Laplace Approximation*

- Gaussian prior, likelihood ⇒ **Gaussian posterior:**
  \[ \pi_{\text{pos}}(\mathbf{z} | \mathbf{y}^{\text{obs}}) = N(\mathbf{z}_{\text{MAP}}, \Gamma_{\text{post}}) \]

- **Linearize** parameter-to-observable map around MAP point:
  \[ \mathbf{y}^{\text{obs}} = f(\mathbf{z}) + \epsilon \approx f(\mathbf{z}_{\text{MAP}}) + F(\mathbf{z} - \mathbf{z}_{\text{MAP}}) + \epsilon \]

- **Covariance** of Gaussian **posterior** given by:
  \[
  \Gamma_{\text{post}} = \left( \tilde{H}_{\text{misfit}}^{\text{PCE}} + \Gamma_{\text{prior}}^{-1} \right)^{-1}
  \]

**Symbols***:
- \( V_r, D_r \): eigenvecs, eigenvals of \( \tilde{H}_{\text{misfit}} \)
- \( \tilde{H}_{\text{misfit}} = \text{prior-preconditioned Hessian} \)
- of data misfit = \( \Gamma_{\text{prior}}^{1/2} H_{\text{misfit}} \Gamma_{\text{prior}}^{1/2} \)
- \( H_{\text{misfit}} = \text{Gauss-Newton portion of} \)
- Hessian misfit = \( F^\dagger \Gamma_{\text{obs}}^{-1} F \)
- \( \tilde{V}_r = \Gamma_{\text{prior}}^{1/2} V_r, \tilde{V}_r^{\dagger} = \text{adjoint of} \tilde{V}_r \)
- \( \Gamma_{\text{prior}}^{-1} = M^{-1}K, K = \text{Laplace stiffness} \).

Bayesian Inference

**Approach 2:** Normal Approximation + Low Rank Laplace Approximation*

- Gaussian prior, likelihood ⇒ *Gaussian posterior:* 
  \[ \pi_{\text{pos}}(\mathbf{z} | \mathbf{y}^{\text{obs}}) = N(\mathbf{z}_{\text{MAP}}, \Gamma_{\text{post}}) \]

- *Linearize* parameter-to-observable map around MAP point:
  \[ \mathbf{y}^{\text{obs}} = f(\mathbf{z}) + \epsilon \approx f(\mathbf{z}_{\text{MAP}}) + F(\mathbf{z} - \mathbf{z}_{\text{MAP}}) + \epsilon \]

- *Covariance* of Gaussian posterior given by:
  \[ \Gamma_{\text{post}} = \left( H_{\text{misfit}}^{\text{PCE}} + \Gamma_{\text{prior}}^{-1} \right)^{-1} \]

*Symbols*:

- \( V_r, D_r \): eigenvcs, eigenvals of \( \tilde{H}_{\text{misfit}} \)
- \( \tilde{H}_{\text{misfit}} = \text{prior-preconditioned Hessian of data misfit} = \Gamma_{\text{prior}}^{-1/2} H_{\text{misfit}} \Gamma_{\text{prior}}^{1/2} \)
- \( H_{\text{misfit}} = \text{Gauss-Newton portion of Hessian misfit} = F^h \Gamma_{\text{obs}}^{-1} F \)
- \( \tilde{V}_r = \Gamma_{\text{prior}}^{1/2} V_r, \tilde{V}_r^\diamond = \text{adjoint of } \tilde{V}_r \)
- \( \Gamma_{\text{prior}}^{-1} = M^{-1} K, K = \text{Laplace stiffness.} \)

* Bui-Thanh, Ghattas, Martin, Stadler, SISC, 2013.
Bayesian Inference

**Approach 2:** Normal Approximation + Low Rank Laplace Approximation*

- Gaussian prior, likelihood ⇒ **Gaussian posterior:**
  \[ \pi_{\text{pos}}(z \mid y^{\text{obs}}) = N(z_{\text{MAP}}, \Gamma_{\text{post}}) \]

- **Linearize** parameter-to-observable map around MAP point:
  \[ y^{\text{obs}} = f(z) + \epsilon \approx f(z_{\text{MAP}}) + F(z - z_{\text{MAP}}) + \epsilon \]

- **Covariance** of Gaussian posterior given by:
  \[ \Gamma_{\text{post}} = \left( H_{\text{misfit}}^{\text{PCE}} + \Gamma_{\text{prior}}^{-1} \right)^{-1} \]

  * Dense!

  **Symbols***:
  
  - \( V_r, D_r \): eigenvecs, eigenvals of \( \tilde{H}_{\text{misfit}} \)
  - \( \tilde{H}_{\text{misfit}} = \text{prior-preconditioned Hessian of data misfit} = \Gamma_{\text{prior}}^{1/2} H_{\text{misfit}} \Gamma_{\text{prior}}^{1/2} \)
  - \( H_{\text{misfit}} = \text{Gauss-Newton portion of Hessian misfit} = F^h \Gamma_{\text{obs}}^{-1} F \)
  - \( \tilde{V}_r = \Gamma_{\text{prior}}^{1/2} V_r, \tilde{V}_r^{\dagger} = \text{adjoint of } \tilde{V}_r \)
  - \( \Gamma_{\text{prior}}^{-1} = M^{-1} K, K = \text{Laplace stiffness.} \)

Bayesian Inference

**Approach 2:** Normal Approximation + Low Rank Laplace Approximation*

- Gaussian prior, likelihood ⇒ **Gaussian posterior:**
  \[ \pi_{\text{pos}}(z \mid y^{\text{obs}}) = N(z_{\text{MAP}}, \Gamma_{\text{post}}) \]

- **Linearize** parameter-to-observable map around MAP point:
  \[ y^{\text{obs}} = f(z) + \epsilon \approx f(z_{\text{MAP}}) + F(z - z_{\text{MAP}}) + \epsilon \]

- **Covariance** of Gaussian posterior given by:
  \[ \Gamma_{\text{post}} = (H_{\text{misfit}}^{\text{PCE}} + \Gamma_{\text{prior}}^{-1})^{-1} \]
  Dense!

- **Low-rank approximation** of \( \Gamma_{\text{post}} \) obtained using Sherman-Morrison-Woodbury formula:
  \[ \Gamma_{\text{post}} \approx \Gamma_{\text{prior}} - \tilde{V}_r D_r \tilde{V}_r \]

**Symbols***:
- \( V_r, D_r \): eigenvvecs, eigenvals of \( \tilde{H}_{\text{misfit}} \)
- \( \tilde{H}_{\text{misfit}} = \) prior-preconditioned Hessian of data misfit = \( \Gamma_{\text{prior}}^{1/2} H_{\text{misfit}} \Gamma_{\text{prior}}^{1/2} \)
- \( H_{\text{misfit}} = \) Gauss-Newton portion of Hessian misfit = \( F^\dagger \Gamma_{\text{obs}}^{-1} F \)
- \( \tilde{V}_r = \Gamma_{\text{prior}}^{1/2} V_r, \tilde{V}_r ^\dagger = \) adjoint of \( \tilde{V}_r \)
- \( \Gamma_{\text{prior}}^{-1} = M^{-1} K, K = \) Laplace stiffness.

* Bui-Thanh, Ghattas, Martin, Stadler, SISC, 2013.
Bayesian Inference

**Approach 2:** Normal Approximation + Low Rank Laplace Approximation*

- **Gaussian prior, likelihood ⇒ Gaussian posterior:**
  \[
  \pi_{\text{pos}}(z | y^{\text{obs}}) = N(z_{\text{MAP}}, \Gamma_{\text{post}})
  \]

- **Linearize** parameter-to-observable map around MAP point:
  \[
  y^{\text{obs}} = f(z) + \epsilon \approx f(z_{\text{MAP}}) + F(z - z_{\text{MAP}}) + \epsilon
  \]

- **Covariance** of Gaussian posterior given by:
  \[
  \Gamma_{\text{post}} = \left( \tilde{H}_{\text{misfit}}^{\text{PCE}} + \Gamma_{\text{prior}}^{-1} \right)^{-1}
  \]
  Dense!

- **Low-rank approximation** of \( \Gamma_{\text{post}} \) obtained using Sherman-Morrison-Woodbury formula:
  \[
  \Gamma_{\text{post}} \approx \Gamma_{\text{prior}} - \tilde{V}_r D_r \tilde{V}_r^\diamond
  \]

**Symbols***:

- \( V_r, D_r \): eigenvecs, eigenvals of \( \tilde{H}_{\text{misfit}} \)
- \( \tilde{H}_{\text{misfit}} = \text{prior-preconditioned Hessian of data misfit} = \Gamma_{\text{prior}}^{1/2} H_{\text{misfit}} \Gamma_{\text{prior}}^{1/2} \)
- \( H_{\text{misfit}} = \text{Gauss-Newton portion of Hessian misfit} = F^\dagger \Gamma_{\text{obs}}^{-1} F \)
- \( \tilde{V}_r = \Gamma_{\text{prior}}^{1/2} V_r, \tilde{V}_r^\diamond = \text{adjoint of } \tilde{V}_r \)
- \( \Gamma_{\text{prior}}^{-1} = M^{-1} K, K = \text{Laplace stiffness.} \)

Bayesian Inference

**Approach 2:** Normal Approximation + Low Rank Laplace Approximation*

- **Gaussian prior, likelihood ⇒ Gaussian posterior:**
  \[ \pi_{pos}(z | y^{obs}) = N(z_{MAP}, \varGamma_{post}) \]

- **Linearize** parameter-to-observable map around MAP point:
  \[ y^{obs} = f(z) + \epsilon \approx f(z_{MAP}) + F(z - z_{MAP}) + \epsilon \]

- **Covariance** of Gaussian posterior given by:
  \[ \varGamma_{post} = (\varGamma_{prior}^{-1} + \varGamma_{misfit}^PCE)^{-1} \]
  Dense!

- **Low-rank approximation** of \( \varGamma_{post} \) obtained using Sherman-Morrison-Woodbury formula:
  \[ \varGamma_{post} \approx \varGamma_{prior} - \tilde{V}_r D_r \tilde{V}_r^{\diamond} \]

- \( \tilde{H}_{misfit} \) and its eigenvalue decomposition can be computed efficiently using a parallel
  **matrix-free Lanczos method.**

- **Rank** \( \varGamma_{post} \) = # modes informing directions of posterior (active subspace vectors**).


**Symbols*:**

- \( V_r, D_r \): eigenvecs, eigenvals of \( \tilde{H}_{misfit} \)
- \( \tilde{H}_{misfit} = \) prior-preconditioned Hessian of data misfit = \( \varGamma_{prior}^{1/2} H_{misfit} \varGamma_{prior}^{1/2} \)
- \( H_{misfit} = \) Gauss-Newton portion of Hessian misfit = \( F^h \varGamma_{obs}^{-1} F \)
- \( \tilde{V}_r = \varGamma_{prior}^{1/2} V_r \), \( \tilde{V}_r^{\diamond} = \) adjoint of \( \tilde{V}_r \)
- \( \varGamma_{prior}^{-1} = M^{-1} K \), \( K = \) Laplace stiffness.
**Bayesian Inference**

**Approach 2:** Normal Approximation + Low Rank Laplace Approximation*

**Upshots:**

😊 Eigenvalues of prior-preconditioned misfit Hessian $\tilde{H}_{\text{misfit}}$ decay rapidly and decay is independent of # parameters.

---

* Figures above: eigenvalue decay of prior preconditioned misfit Hessian

---

* *Bui-Thanh, Ghattas, Martin, Stadler, SISC, 2013.*
Bayesian Inference

**Approach 2:** Normal Approximation + Low Rank Laplace Approximation*

**Upshots:**

😊 Prior preconditioned misfit *eigenvectors* have *physical interpretation*:
- First modes correspond to regions which are *highly informed by data*
- Modes become more *global* as eigenvalues decay

---

Mode 1  Mode 2  Mode 3  Mode 200
Bayesian Inference

**Approach 2:** Normal Approximation + Low Rank Laplace Approximation*

**Upshots:**
- ☺ Prior preconditioned misfit *eigenvectors* have **physical interpretation**:
  - First modes correspond to regions which are *highly informed by data*
  - Modes become more *global* as eigenvalues decay
- ☻ The use of data has **dramatically reduces** the *posterior variance*
Bayesian Inference

**Approach 2:** Normal Approximation + Low Rank Laplace Approximation*

**Issues:**

⚠️ PDF will be **Gaussian** – general PDFs cannot be obtained.
Bayesian Inference

**Approach 2:** Normal Approximation + Low Rank Laplace Approximation*

**Issues:**

- PDF will be **Gaussian** – general PDFs cannot be obtained.

- Laplace equation (regularization) **involves correlation length parameter** that changes decay of eigenvalues of prior preconditioned Hessian.
Bayesian Inference

**Approach 2:** Normal Approximation + Low Rank Laplace Approximation*

**Issues:**

- PDF will be **Gaussian** – general PDFs cannot be obtained.

- Laplace equation (regularization) **involves correlation length parameter** that changes decay of eigenvalues of prior preconditioned Hessian.

- Dimension of parameter space is **too high** $O(1000)$ for forward propagation.
Bayesian Inference

**Approach 2:** Normal Approximation + Low Rank Laplace Approximation*

**Issues:**

- PDF will be **Gaussian** – general PDFs cannot be obtained.

- Laplace equation (regularization) **involves correlation length parameter** that changes decay of eigenvalues of prior preconditioned Hessian.

- **Dimension** of parameter space is **too high** $O(1000)$ for forward propagation.

- Log-normal prior may be cause of (nonphysical) **bias** towards mass increase when performing **forward propagation**.
Bayesian Inference

**Ongoing work:**

- Use **low fidelity** models (e.g. SIA) to study problems (such as bias in SLR on previous slide) with the large-scale, high-resolution, expensive end-to-end framework.

- Use dimension reduction, leveraging **transient adjoints** obtained from new model suite, to reduce cost of **propagating uncertainties** through **transient model**.

**Dimension reduction** by **adding physics**: subglacial hydrology models rely on only a handful of parameters that, to first approximation, can be considered uniform

\[
\beta(u) = \mu_f N \left( \frac{|u|}{|u| + \lambda AN^n} \right)^q \frac{1}{|u|} + \text{Thickness equation (subglacial hydrology)}
\]

*Figure 1*: ISMIP-HOM B test + SIA and BP models is \(>1000\times\) less than GIS.

*Figure 2*: gradients can determine directions that significantly impact SLR.
MPI+X FEA via Kokkos

• **MPI-only** nested for loop:

```c
for (int cell=0; cell<numCells; ++cell) 
    for (int node=0; node<numNodes; ++node) 
        for (int qp=0; qp<numQPs; ++qp) 
            compute A;            MPI process n
```
MPI+X FEA via Kokkos

- **Multi-dimensional parallelism** for nested for loops via Kokkos:
  
  ```
  for (int cell=0; cell<numCells; ++cell)
    for (int node=0; node<numNodes; ++node)
      for (int qp=0; qp<numQPs; ++qp)
        compute A;  
  ```

  MPI process $n$

  Thread 1 computes A for $(cell, node, qp) = (0, 0, 0)$

  Thread 2 computes A for $(cell, node, qp) = (0, 0, 1)$

  : 

  Thread $N$ computes A for $(cell, node, qp) = (numCells, numNodes, numQPs)$
MPI+X FEA via *Kokkos*

- **Multi-dimensional parallelism** for nested for loops via *Kokkos*:

```cpp
for (int cell=0; cell<numCells; ++cell) 
    for (int node=0; node<numNodes; ++node) 
        for (int qp=0; qp<numQPs; ++qp) 
            compute A;  

Kokkos::Experimental::md_parallel_for<ExecutionSpace>(range,*this);
```

*Unified Virtual Memory.*
MPI+X FEA via Kokkos

- **Multi-dimensional parallelism** for nested for loops via Kokkos:

  ```c
  for (int cell=0; cell<numCells; ++cell)
      for (int node=0; node<numNodes; ++node)
          for (int qp=0; qp<numQPs; ++qp)
              compute A;
  ```

  Thread 1 computes A for (cell,node,qp)=(0,0,0)

  Thread 2 computes A for (cell,node,qp)=(0,0,1)

  Thread N computes A for (cell,node,qp)=(numCells,numNodes,numQPs)

  ```c
  computeA_Policy range({0,0,0},{(int)numCells,(int)numNodes,(int)numQPs});
  Kokkos::Experimental::md_parallel_for<ExecutionSpace>(range,*this);
  ```

- **ExecutionSpace** defined at *compile time*, e.g.

  ```c
  typedef Kokkos::OpenMP ExecutionSpace; //MPI+OpenMP
  typedef Kokkos::CUDA ExecutionSpace; //MPI+CUDA
  typedef Kokkos::Serial ExecutionSpace; //MPI-only
  ```
MPI+X FEA via *Kokkos*

- **Multi-dimensional parallelism** for nested for loops via *Kokkos*:

  ```
  for (int cell=0; cell<numCells; ++cell)
  for (int node=0; node<numNodes; ++node)
  for (int qp=0; qp<numQPs; ++qp)
      compute A;
  ```

  Thread 1 computes A for \((cell, node, qp)=(0,0,0)\)

  Thread 2 computes A for \((cell, node, qp)=(0,0,1)\)

  \vdots

  Thread N computes A for \((cell, node, qp)=(numCells, numNodes, numQPs)\)

- **ExecutionSpace** defined at **compile time**, e.g.
  ```
  typedef Kokkos::OpenMP ExecutionSpace; //MPI+OpenMP
  typedef Kokkos::CUDA ExecutionSpace; //MPI+CUDA
  typedef Kokkos::Serial ExecutionSpace; //MPI-only
  ```

- **Atomics** used to scatter local data to global data structures
  ```
  Kokkos::atomic_fetch_add
  ```

  ```
  computeA_Policy range({0,0,0},{(int)numCells,(int)numNodes,(int)numQPs});
  Kokkos::Experimental::md_parallel_for<ExecutionSpace>(range,*this);
  ```
MPI+X FEA via Kokkos

- **Multi-dimensional parallelism** for nested for loops via Kokkos:

  ```
  for (int cell=0; cell<numCells; ++cell)
    for (int node=0; node<numNodes; ++node)
      for (int qp=0; qp<numQPs; ++qp)
        compute A;
  ```

- **ExecutionSpace** defined at *compile time*, e.g.
  ```
  typedef Kokkos::OpenMP ExecutionSpace; //MPI+OpenMP
  typedef Kokkos::CUDA ExecutionSpace; //MPI+CUDA
  typedef Kokkos::Serial ExecutionSpace; //MPI-only
  ```

- **Atomics** used to scatter local data to global data structures
  ```
  Kokkos::atomic_fetch_add
  ```

- For MPI+CUDA, data transfer from host to device handled by **CUDA UVM**.

* Unified Virtual Memory.
MPI+X FEA via *Kokkos*

- **Multi-dimensional parallelism** for nested for loops via *Kokkos*:

  ```cpp
  for (int cell=0; cell<numCells; ++cell)
    for (int node=0; node<numNodes; ++node)
      for (int qp=0; qp<numQPs; ++qp)
        compute A;
  ```

  
  Thread 1 computes A for (cell,node,qp)=(0,0,0)
  
  Thread 2 computes A for (cell,node,qp)=(0,0,1)
  
  \vdots
  
  Thread N computes A for (cell,node,qp)=(numCells,numNodes,numQPs)

- ExecutionSpace defined at **compile time**, e.g.

  ```cpp
  typedef Kokkos::OpenMP ExecutionSpace; //MPI+OpenMP
  typedef Kokkos::CUDA ExecutionSpace; //MPI+CUDA
  typedef Kokkos::Serial ExecutionSpace; //MPI-only
  ```

- **Atomics** used to scatter local data to global data structures
  
  ```cpp```
  Kokkos::atomic_fetch_add
  ```

- For MPI+CUDA, data transfer from host to device handled by **CUDA UVM**.

  * Unified Virtual Memory.
$\frac{\partial H}{\partial t} = -\nabla \cdot (\bar{u} H) + \dot{b}$

$\frac{\rho c}{\partial t} = \nabla \cdot (k \nabla T) - \rho c u \cdot \nabla T + 2 \dot{e} \sigma$

\[ \begin{align*}
-\nabla \cdot (2 \mu \varepsilon_1) &= -\rho g \frac{\partial s}{\partial x} \\
-\nabla \cdot (2 \mu \varepsilon_2) &= -\rho g \frac{\partial s}{\partial y}
\end{align*} \]

“Loose” sequential/staggered coupling between $u$ and $(T, H)$. 
**FO Stokes-Thickness Coupling**

- **MPAS Land-Ice (Fortran)**
  - Thickness evolution, temperature solve, coupling to DOE-ESM

- **C++/Fortran Interface, Mesh Conversion**

- **Albany Land-Ice (C++)**
  - velocity solve

**Equations**

\[
\frac{\partial H}{\partial t} = -\nabla \cdot (\bar{u}H) + \dot{b}
\]

*H equation is solved with upwind scheme + incremental remap.*

\[
\begin{align*}
-\nabla \cdot (2\mu \dot{e}_1) &= -\rho g \frac{\partial s}{\partial x} \\
-\nabla \cdot (2\mu \dot{e}_2) &= -\rho g \frac{\partial s}{\partial y}
\end{align*}
\]
FO Stokes-Thickness Coupling

**MPAS Land-Ice (Fortran)**
Thickness evolution, temperature solve, coupling to DOE-ESM

**C++/Fortran Interface, Mesh Conversion**

**Albany Land-Ice (C++)**
velocity solve

**LandIce_model**

**output file**

\[ \frac{\partial H}{\partial t} = -\nabla \cdot (\bar{u}H) + \dot{b} \]

*H equation is solved with upwind scheme + incremental remap.*

\[
\begin{align*}
-\nabla \cdot (2\mu \varepsilon_1) &= -\rho g \frac{\partial s}{\partial x} \\
-\nabla \cdot (2\mu \varepsilon_2) &= -\rho g \frac{\partial s}{\partial y}
\end{align*}
\]

😊 **Upside:** scheme fits nicely into existing codes
FO Stokes-Thickness Coupling

**LandIce_model**

- **MPAS Land-Ice (Fortran)**
  - Thickness evolution, temperature solve, coupling to DOE-ESM

  \[
  \frac{\partial H}{\partial t} = -\nabla \cdot (\overline{u} H) + \dot{b}
  \]
  - H equation is solved with upwind scheme + incremental remap.

- **C++/Fortran Interface, Mesh Conversion**

- **Albany Land-Ice (C++)**
  - velocity solve

\[
\begin{align*}
-\nabla \cdot (2\mu \dot{e}_1) &= -\rho g \frac{\partial s}{\partial x} \\
-\nabla \cdot (2\mu \dot{e}_2) &= -\rho g \frac{\partial s}{\partial y}
\end{align*}
\]

- **Upside:** scheme fits nicely into existing codes

- **Downside:** for problems with shallow ice on frozen bedrock, need to satisfy very restrictive **diffusive CFL** condition*:
  \[\Delta t \leq CFL_{\text{diff}}(\Delta x)^2\]
FO Stokes-Thickness Coupling

\[ \frac{\partial H}{\partial t} = -\nabla \cdot (\bar{u} H) + \dot{b} \]

**H equation is solved with upwind scheme + incremental remap.**

**Upside:** scheme fits nicely into existing codes

**Downside:** for problems with shallow ice on frozen bedrock, need to satisfy very restrictive **diffusive CFL condition**: \( \Delta t \leq CFL_{\text{diff}} (\Delta x)^2 \)

**Downside:** Very crude representation of ice advancement/retreat
Semi-Implicit Coupling

Unstructured explicit finite volume on Voronoi grids
Solves for thickness (upwind method)

$H$

Unstructured finite element

- MPAS computes thickness $H$, uses it to define geometry, which is passed to ALI.
Semi-Implicit Coupling

**MPAS**
Model for Prediction Across Scales

Unstructured **explicit** finite volume on Voronoi grids
Solves for **thickness** (upwind method)

**Albany**
Unstructured finite element
Solves FO Stokes for **velocity-thickness** together

- MPAS computes thickness $H$, uses it to define geometry, which is passed to ALI.
- ALI computes coupled velocity-thickness $(u, H)$ pair:

$$-2 \mu (u^{(n+1)}) \nabla \cdot \varepsilon(u^{(n+1)}) = -\rho g \nabla (b + H^{(n+1)}), \quad \text{in } \Omega_{H^{(n+1)}}$$

$$\frac{H^{(n+1)} - H^{(n)}}{\Delta t} = -\nabla \cdot (\overline{u}^{(n+1)}H^{(n+1)}) + \dot{b}$$

**Idea:** the velocity computed by the coupled system FO-thickness equation will be **more stable** than the one computed by FO Stokes only and will allow use of larger $\Delta t$. 

Semi-Implicit Coupling

MPAS computes thickness $H$, uses it to define geometry, which is passed to ALI.

ALI computes coupled velocity-thickness $(u, H)$ pair:

$$
-2\mu(u^{(n+1)})\nabla \cdot \dot{\varepsilon}(u^{(n+1)}) = -\rho g \nabla (b + H^{(n+1)}), \quad \text{in } \Omega_{H^{(n+1)}}
$$

$$
\frac{H^{(n+1)} - H^{(n)}}{\Delta t} = -\nabla \cdot (\overline{u}^{(n+1)}H^{(n+1)}) + \dot{b}
$$

Idea: the velocity computed by the coupled system FO-thickness equation will be more stable than the one computed by FO Stokes only and will allow use of larger $\Delta t$.

Only velocity $u$ is passed back to MPAS.
Semi-Implicit Coupling

MPAS computes thickness $H$, uses it to define geometry, which is passed to ALI.

ALI computes coupled velocity-thickness $(\mathbf{u}, H)$ pair:

$$
-2\mu(\mathbf{u}^{(n+1)})\nabla \cdot \mathbf{\epsilon}(\mathbf{u}^{(n+1)}) = -\rho g \nabla (b + H^{(n+1)}), \quad \text{in } \Omega_{H^{(n+1)}}
$$

$$
\frac{H^{(n+1)} - H^{(n)}}{\Delta t} = -\nabla \cdot (\overline{\mathbf{u}}^{(n+1)} H^{(n+1)}) + \mathbf{b}
$$

Idea: the velocity computed by the coupled system FO-thickness equation will be more stable than the one computed by FO Stokes only and will allow use of larger $\Delta t$

Only velocity $\mathbf{u}$ is passed back to MPAS.

Downside: more intrusive implementation; larger system; expense associated to geometry changing between iterations (use Newton to compute shape derivatives).
Semi-Implicit Approach: Dome Test Case

Top left: reference solution computed using sequential approach and time step of 5 months

Semi-implicit approach allows the use of much larger time-steps than sequential approach!
Semi-Implicit Approach: Antarctica

- Variable-resolution Antarctica grid with maximum resolution of 3km.
- Compared **semi-implicit** with adaptive $\Delta t$ based on **advective CFL condition** vs. **explicit** scheme based on **diffusive CFL condition**.
- **Sequential approach:** $\Delta t = O(\text{days})$
- **Semi-Implicit approach:** $\Delta t = O(\text{months})$
- **Cost of iteration** is **larger** for semi-implicit scheme because of increased dimension of nonlinear system (more expensive assembly and solve).
- Nonetheless, with semi-implicit scheme, we obtained **speedup of 4.5×** (~2 year run).

*Basal friction:* obtained with inversion.

*Geometry:* Bedmap2 (Fretwell et al., Cryosphere, 2013), managed by D. Martin and X. Asay-Davis.

*Temperature:* Cornford, Martin et al, 2014; Pattyn et al., 2010.

*Mesh:* unstructured Delaynay mesh refined based on surface velocity (MPAS planar Voronoi grid generator by M. Duda, NCAR).
Towards Fully Implicit FO Stokes-Thickness Coupling

- We are looking at the following **fully implicit** formulations:
  - **Level set** formulation coupled with the thickness evolution equation is used to track the front position*: no need to modify mesh, can handle changes in topography.
  - Thickness equation as an **obstacle problem/variational inequality****: no need to track boundary, amenable to implicit integration

\[
\frac{\partial H}{\partial t} = -\nabla \cdot (\bar{u}H) + \dot{b}, \quad \text{in } \Sigma^+
\]

\[
\int \frac{\partial H}{\partial t} (v - H) \geq \int (\bar{u}H) \cdot \nabla (v - H) + \int \theta(v - H), \quad H \geq 0, \forall v \geq 0, \text{ in } \Sigma
\]

PISCEES & E3SM Coupling Validation

Sub-shelf melt rates (RRS30to10km resolution)

* Rignot et al., Science, 2013