Albany Land-Ice (ALI): A Next-Generation Variable-Resolution Ice Sheet Model Towards Probabilistic Projections of Sea-Level Change

Irina K. Tezaur

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

Seminar Uppsala University October 17, 2019

Acknowledgements


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
   - ALI steady stress-velocity solver
     ➢ Discretization & meshes
     ➢ Nonlinear solvers
     ➢ Linear solvers & parallelization
     ➢ Performance-portability
     ➢ Ice sheet initialization
   - MALI for dynamic simulations
     ➢ Velocity-thickness/temperature coupling
     ➢ Towards science runs & UQ

3. 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
   - ALI steady stress-velocity solver
     - Discretization & meshes
     - Nonlinear solvers
     - Linear solvers & parallelization
     - Performance-portability
     - Ice sheet initialization
   - MALI for dynamic simulations
     - Velocity-thickness/temperature coupling
     - Towards science runs & UQ

3. Ongoing & future work
Motivation

- Global mean sea-level is rising at the rate of **3.2 mm/year** and the rate is **increasing**!
- Latest studies suggest possible increase in sea-level of **0.3-2.5m** by 2100.
- **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

➢ Global mean sea-level is rising at the rate of 3.2 mm/year and the rate is increasing!
➢ Latest studies suggest possible increase in sea-level of 0.3-2.5m by 2100.
➢ **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.*
Outline

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

2. Algorithms and software
   • ALI steady stress-velocity solver
     ➢ Discretization & meshes
     ➢ Nonlinear solvers
     ➢ Linear solvers & parallelization
     ➢ Performance-portability
     ➢ Ice sheet initialization
   • MALI for dynamic simulations
     ➢ Velocity-thickness/temperature coupling
     ➢ Towards science runs & UQ

3. 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 ESM has 6 modular components:

1. Atmosphere model
2. Ocean model
3. Sea ice model
4. Land ice model
5. Land model
6. Flux coupler

**Goal of ESM:** to provide actionable scientific predictions of 21st century sea-level change (including uncertainty bounds).
An ESM has 6 modular components:

1. Atmosphere model
2. Ocean model
3. Sea ice model
4. Land ice model
5. Land model
6. Flux coupler

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

An ESM has **6 modular components**:

1. Atmosphere model
2. Ocean model
3. Sea ice model
4. Land ice model
5. Land model
6. 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.

“Although ice sheet models have improved in recent years, much work is needed to make these models robust and efficient on continental scales and to quantify uncertainties in their projected outputs”.
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.

“Although ice sheet models have improved in recent years, much work is needed to make these models robust and efficient on continental scales and to quantify uncertainties in their projected outputs”.

DOE-funded Land-Ice Modeling Projects:


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.

“Although ice sheet models have improved in recent years, much work is needed to make these models robust and efficient on continental scales and to quantify uncertainties in their projected outputs”.

DOE-funded Land-Ice Modeling Projects:


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)

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

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

Two land-ice dycores currently under development

**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)

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

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

Two land-ice dycores currently under development

**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**
   - ALI steady stress-velocity solver
     - Discretization & meshes
     - Nonlinear solvers
     - Linear solvers & parallelization
     - Performance-portability
     - Ice sheet initialization
   - MALI for dynamic simulations
     - Velocity-thickness/temperature coupling
     - Towards science runs & UQ

3. **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*: \( \mu = \frac{1}{2} A(T) \frac{1}{n} \left( \frac{1}{2} \sum_{ij} \varepsilon_{ij}^2 \right)^{\frac{1}{2n-1}} \)
- 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**.

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

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

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

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

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

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

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

$\mathbf{\epsilon}(\mathbf{u}) = \left( \begin{array}{ccc}
    u_x & \frac{1}{2}(u_y + v_x) & \frac{1}{2}(u_z + w_x) \\
    \frac{1}{2}(u_y + v_x) & v_y & \frac{1}{2}(v_z + w_y) \\
    \frac{1}{2}(u_z + w_x) & \frac{1}{2}(v_z + w_y) & w_z
    \end{array} \right)$

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

*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

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

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.

\[ u \equiv (u, v, w) \]

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

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

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

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

*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:**

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

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].”
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 $T$.

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

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

  \[ \bar{\mathbf{u}} = \text{vertically averaged} \ \mathbf{u} \]
  \[ \dot{b} = \text{surface mass balance} \]

  *(given accumulation-ablation function that accounts for e.g. accumulation due to snowfall)*

  $\Gamma$ = horizontal extent of the ice

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

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

  ⬤ time $t_0$
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 $T$.

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

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

- $\bar{\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.

Time $t_1$

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 $T$.

- **Thickness equation** for the ice thickness $H$:
  \[
  \frac{\partial H}{\partial t} + \nabla \cdot (\bar{\mathbf{u}} H) + \dot{b}, \quad \text{on} \quad \Gamma
  \]
  - $\bar{\mathbf{u}} = \text{vertically averaged } \mathbf{u}$
  - $\dot{b} = \text{surface mass balance}$
  - *(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
   • Earth system models & projects
   • Land-ice equations
   • Our codes: ALI, MALI

2. Algorithms and software
   • ALI steady stress-velocity solver
     ➢ Discretization & meshes
     ➢ Solvers & preconditioners
     ➢ Performance-portability
     ➢ Ice sheet initialization
   • MALI for dynamic simulations
     ➢ Velocity-thickness/temperature coupling
     ➢ Towards science runs & UQ

3. Ongoing & future work

4. Summary
Our Codes

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

\[
\begin{align*}
-\nabla \cdot (2\mu \epsilon_1) &= -\rho g \frac{\partial s}{\partial x}, \text{ in } \Omega \\
-\nabla \cdot (2\mu \epsilon_2) &= -\rho g \frac{\partial s}{\partial y}
\end{align*}
\]

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

**Conservation of Mass: thickness** evolution PDE

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

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

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

*https://github.com/gahansen/Albany.

**Codes:**

- **MAMAL = MPAS + ALI**
- **Albany** = multi-physics PDE code
- **Albany Land-Ice (ALI)**
- **Trilinos**
- **MPAS** Model for Prediction Across Scales
- **E3SM** Energy Exascale Earth System Model

* https://github.com/gahansen/Albany.

---

*Velocity solve is most expensive!*
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

**Trilinos**: [https://github.com/trilinos/Trilinos](https://github.com/trilinos/Trilinos)

**Dakota**: [https://dakota.sandia.gov/](https://dakota.sandia.gov/)

**Albany**: [https://github.com/SNL Computation/Albany](https://github.com/SNL Computation/Albany)
Model for Prediction Across Scales (MPAS): climate modeling framework built around SCVT* meshes (LANL + NCAR collaboration)

*SCVT = Spherical Centroidal Voronoi Tessellations

- 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)
Outline

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

2. Algorithms and software
   - ALI steady stress-velocity solver
     - Discretization & meshes
     - Nonlinear solvers
     - Linear solvers & parallelization
     - Performance-portability
     - Ice sheet initialization
   - MALI for dynamic simulations
     - Velocity-thickness/temperature coupling
     - Towards science runs & UQ

3. 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

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

**MALI** uses dual of hexagonal mesh extruded to tetrahedra.

**Variable resolution triangular mesh extruded to a (thin) tetrahedral mesh.**
Meshes

- 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.

**MALI** uses dual of hexagonal mesh extruded to tetrahedra.

Variable resolution triangular mesh extruded to a (thin) tetrahedral mesh.
Meshes

- 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.

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

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

2. Algorithms and software
   • **ALI steady stress-velocity solver**
     ➢ Discretization & meshes
     ➢ **Nonlinear solvers**
     ➢ Linear solvers & parallelization
     ➢ Performance-portability
     ➢ Ice sheet initialization
   • **MALI for dynamic simulations**
     ➢ Velocity-thickness/temperature coupling
     ➢ Towards science runs & UQ

3. Ongoing & future work
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.

*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:

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

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

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

*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:

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

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

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

*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.  

Improved **robustness** and **faster** nonlinear convergence by doing a **homotopy continuation** w.r.t. $\gamma$

Glen’s Law Viscosity:

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

$\gamma$ = regularization parameter

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

*Tezaur et al. 2015.*
1. **Background**
   - Motivation for climate & land-ice modeling
   - ISMs, ESMs & projects
   - Land-ice equations
   - Our codes: ALI, MALI

2. **Algorithms and software**
   - ALI steady stress-velocity solver
     - Discretization & meshes
     - Nonlinear solvers
     - **Linear solvers & parallelization**
     - Performance-portability
     - Ice sheet initialization
   - MALI for dynamic simulations
     - Velocity-thickness/temperature coupling
     - Towards science runs & UQ

3. **Ongoing & future work**
From Nonlinear Solvers to Linear Solvers

• **Krylov iterative linear solvers** are employed – CG or GMRES.
  ➢ FO Stokes equations are **symmetric**.
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).
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 **algebraic multi-grid (AMG)** solver has been developed to deal with these issues and is available in Trilinos.
  - **Graph-based algorithms for removing islands/ice hinges** are being developed*.

*Tuminaro et al. 2016.
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 **algebraic multi-grid (AMG)*** solver has been developed to deal with these issues and is available in Trilinos.
  - **Graph-based algorithms** for **removing islands/ice hinges** are being developed*.

*Tuminaro et al. 2016.
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.

\[
A_3 u_3 = f_3
\]

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 $A_2 u_2 = f_2$. 

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.
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

$\times 8^4$ 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!
Moderate Resolution Antarctica Weak Scaling Study

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

- **Along ice shelf front**: open-ocean BC (Neumann).
- **Along ice shelf base**: zero traction BC (Neumann).

⇒ For vertical grid lines that lie within ice shelves, top and bottom BCs resemble Neumann BCs so sub-matrix associated with one of these lines is almost* singular.

(vertical > horizontal coupling) +

Neumann BCs =

nearly singular submatrix associated with vertical lines

⇒ Ice shelves give rise to severe ill-conditioning of linear systems!

*Completely singular in the presence of islands and some ice tongues.
Moderate Resolution Antarctica Weak Scaling Study

- Weak scaling study on Antarctic problem (8km w/ 5 layers → 2km w/ 20 layers).
- Initialized with realistic basal friction (from deterministic inversion) and temperature field from BEDMAP2.
- **Iterative linear solver**: GMRES.
- **Preconditioner**: ILU vs. new AMG based on aggressive semi-coarsening.

**Severe ill-conditioning caused by ice shelves!**

![Graph showing performance comparison between ILU and AMG preconditioners](image)

- **ILU**
  - AMG preconditioner less sensitive than ILU to ill-conditioning (ice shelves → Green’s function* with modest horizontal decay → ILU is less effective).

Outline

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

2. Algorithms and software
   • **ALI steady stress-velocity solver**
     ➢ Discretization & meshes
     ➢ Nonlinear solvers
     ➢ Linear solvers & parallelization
     ➢ **Performance-portability**
     ➢ Ice sheet initialization
   • **MALI for dynamic simulations**
     ➢ Velocity-thickness/temperature coupling
     ➢ Towards science runs & UQ

3. 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, NVIDIA GPU, 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.)
Targeted Computer Architectures/Results

**Cori** (NERSC): 2,388 Haswell nodes [2 Haswell (32 cores)]
9,688 KNL nodes [1 Xeon Phi KNL (68 cores)]

**Summit** (ORLCF): 4600 nodes [2 P9 (22 cores) + V100 (6 GPUs)]

**Aurora** (ALCF): U.S.’s first exascale supercomputer (ETA: 2021)
New Intel Xeon Phi Processor (Knights Hill cancelled)

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

**Targeted Computer Architectures/Results**

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

**Cori** (NERSC): 2,388 Haswell nodes [2 Haswell (32 cores)]
9,688 KNL nodes [1 Xeon Phi KNL (68 cores)]

**Summit** (ORLCF): 4600 nodes [2 P9 (22 cores) + V100 (6 GPUs)]

**Aurora** (ALCF): U.S.’s first exascale supercomputer (ETA: 2021)
New Intel Xeon Phi Processor (Knights Hill cancelled)

**Node Configuration**
- a: 32MPI
- b: 32(MPI+OMP)
- c: 68MPI
- d: 68(MPI+4OMP)
- e: 16MPI
- f: 4(MPI+GPU)

**MPI+X Single-Node Speedups**

**MPI+X strong-scaling**

**Wall-clock time (s)**

- **Cori** (Haswell): 1.2x
- **Cori** (KNL) Clusters: 1.2x
- **Ride** (POWER8,P100): 12.6x
Outline

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

2. Algorithms and software
   - **ALI steady stress-velocity solver**
     - Discretization & meshes
     - Nonlinear solvers
     - Linear solvers & parallelization
     - Performance-portability
     - **Ice sheet initialization**
   - **MALI for dynamic simulations**
     - Velocity-thickness/temperature coupling
     - Towards science runs & UQ

3. 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.

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

- \(U\): 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*:

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

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

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

- **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*:

\[ \text{minimize } \beta, H \ m(u, H) \]

s.t. FO Stokes PDEs


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

\[
m(u, H) = \int_{\Gamma} \frac{1}{\sigma_u^2} |u - u^{obs}|^2 ds
\]

\[
+ \int_{\Gamma} \frac{1}{\sigma_\tau^2} |\text{div}(UH) - \tau_s|^2 ds
\]

\[
+ \int_{\Gamma} \frac{1}{\sigma_H^2} |H - H^{obs}|^2 ds
\]

\[
+ R(u, H)
\]

- **Surface velocity mismatch**
- **SMB mismatch**
- **Thickness mismatch**
- **Regularization terms**

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

\[ U: \text{computed depth averaged velocity} \]

\[ H: \text{ice thickness} \]

\[ \beta: \text{basal sliding friction coefficient} \]

\[ \tau_s: \text{surface mass balance (SMB)} \]

\[ R(u, H): \text{regularization term} \]

\[ \sigma: \text{standard deviation (weight of uncertainties)} \]
Deterministic Inversion Algorithm & Software

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

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

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>AztecOO+Ifpack/ML</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*

<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Deterministic Inversion: Common vs. Novel Approach*

SMB (m/yr) needed for equilibrium

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

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.

Antarctic ice sheet inversion performed on O(1M) parameters!
Outline

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

2. Algorithms and software
   - ALI steady stress-velocity solver
     - Discretization & meshes
     - Nonlinear solvers
     - Linear solvers & parallelization
     - Performance-portability
     - Ice sheet initialization
   - MALI for dynamic simulations
     - Velocity-thickness/temperature coupling
     - Towards science runs & UQ

3. Ongoing & future work
MPAS + ALI Coupling

LandIce model

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

C++/Fortran Interface, Mesh Conversion

Albany Land-Ice (C++) velocity solve

output file

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

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

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

LandIce_model → MPAS Land-Ice (Fortran) → C++/Fortran Interface, Mesh Conversion → Albany Land-Ice (C++)

output file

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

C++/Fortran Interface, Mesh Conversion

Albany Land-Ice (C++) velocity solve

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

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

$\begin{cases} -\nabla \cdot (2\mu \varepsilon_1) = -\rho g \frac{\partial s}{\partial x} \\ -\nabla \cdot (2\mu \varepsilon_2) = -\rho g \frac{\partial s}{\partial y} \end{cases}$
FO Stokes-Thickness Coupling

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

C++/Fortran Interface, Mesh Conversion

Albany Land-Ice (C++) velocity solve

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

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

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

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

output file

LandIce_model

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

C++/Fortran Interface, Mesh Conversion

Albany Land-Ice (C++)
velocity solve

E3SM
Energy Exascale Earth System Model

MPAS
Model for Prediction Across Scales

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

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

$\begin{cases}
-\nabla \cdot (2\mu \dot{e}_1) = -\rho g \frac{\partial s}{\partial x} \\
-\nabla \cdot (2\mu \dot{e}_2) = -\rho g \frac{\partial s}{\partial y}
\end{cases}$

😊 **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

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

**output file**

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

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

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

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

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

**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 $(\mathbf{u}, H)$ pair:

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

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

**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

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

MPAS

Unstructured finite element
Solves FO Stokes for velocity-thickness together

Albany

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

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

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

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 $(u, H)$ 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.

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.


*Mesh:* unstructured Delaynay mesh refined based on surface velocity (MPAS planar Voronoi grid generator by M. Duda, NCAR).
TowardsFullyImplicit 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

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

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

*Bondzio et al. 2016.*  **Bueler, 2016.**
FO Stokes-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**
  - 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:** \[ u \cdot \nabla h + \nabla \cdot q = \tau : \dot{e} \]
    - **Enthalpy equation:** \[ u \cdot \nabla h + \nabla \cdot q = \tau : \dot{e} \]

- 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:** Computed basal temperature
**Right:** Thawed/frozen map from MacGregor et al., *JGR*, 2016

**Enthalpy/melting graph at bed**

\[ m = m^* \left( \frac{1}{2} + \frac{1}{\pi} \arctan \left( \alpha (h - h_m) \right) \right) \]

\[ m = m^* \left( \frac{1}{2} + \frac{\alpha}{\pi} (h - h_m) \right) \]

\[ \nabla h \cdot n = \rho_w L m - G - \tau_b \cdot u \]
Outline

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

2. Algorithms and software
   • ALI steady stress-velocity solver
     ➢ Discretization & meshes
     ➢ Nonlinear solvers
     ➢ Linear solvers & parallelization
     ➢ Performance-portability
     ➢ Ice sheet initialization
   • MALI for dynamic simulations
     ➢ Velocity-thickness/temperature coupling
     ➢ Towards science runs & UQ

3. Ongoing & future work
Dynamic Simulations: 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*

**ABUMIP = Antarctic BUttressing Model Intercomparison Project**

**Basic idea:** instantaneously remove all ice shelves and see what happens in the next 200 years, preventing any floating ice from ever forming again.

---

Left: 200 year MALI Antarctic ice sheet simulation after instantaneous removal of all floating ice shelves

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

Courtesy of M. Hoffman, S. Price (LANL)

ABUMIP = Antarctic BUttressing Model Intercomparison Project

**Basic idea**: instantaneously remove all ice shelves and see what happens in the next 200 years, preventing any floating ice from ever forming again.

*Left*: simulated Antarctic ice sheet geometry and speed from MALI 200 years after instantaneous removal of all floating ice shelves.

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

*Courtesy of M. Hoffman, S. Price (LANL)*

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.
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

---

**Deterministic inversion**

**Bayesian calibration**

**Forward propagation**

---

\[ \beta, H \text{ PDFs (from Bayesian inference)} \]

\[ \text{SLR}(t) \text{ for ensemble of forward runs with } \beta, H \text{ sampled from its PDF} \]

PDF of SLR

---

* Jakeman *et al.* (in prep), 2018.
Uncertainty Quantification*

**Goal:** obtain PDF of initial condition using Bayesian inference and propagate this PDF through model to get PDF of \textit{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.

**UQ Workflow**

- **Deterministic inversion**
- **Bayesian calibration**
- **Forward propagation**

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

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
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).

Fully **coupled, dynamic ice sheet** simulations will be done through **ProSPect**: awarded ~85M CPU hours at 3 DOE computing centers.

*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
   • ALI steady stress-velocity solver
     • Discretization & meshes
     • Nonlinear solvers
     • Linear solvers & parallelization
     • Performance-portability
     • Ice sheet initialization
   • MALI for dynamic simulations
     • Velocity-thickness/temperature coupling
     • Towards science runs & UQ

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


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. PISCEES 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 PISCEES 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 PISCEES is to create a land-ice solver that is scalable, fast, and robust on continental 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.

Rapid Development of an Ice Sheet Climate Application Using the Components-Based Approach

Andrew Salinger, Jr.
Mauro Pergo
Raymond Tuminaro
Sandia National Labs
Stephen Price
Los Alamos National Labs
Processing hours: 1,000,000

“A computational scientist with expertise in math and algorithms, it is challenging to get deep enough into a new science application area to make an impact. This team has made a sustained effort in learning about ice sheets and building relationships with climate scientists, and has been rewarded seeing our code on the critical path of DOE’s climate science program.”

– Andy Salinger

https://www.sandia.gov/~ikalash
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-Pattyn¹</td>
<td>$O(\delta)$</td>
<td>3D model for $(u_1, u_2)$</td>
<td>continental scale</td>
</tr>
<tr>
<td>L1L1, L1L2²</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

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

**Ice regime:**
grounded ice with frozen bed

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

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

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

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

**Ice regime:**
shelves or fast sliding grounded ice

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

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

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.

**Scorec** = Scientific Computation Research Center at RPI: https://github.com/SCOREC

*PAALS = Parallel Albany Adaptive Loop with SCOREC*

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

Ryder glacier (north coast)

Left: before mesh adaptation; Right: after mesh adaptation
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?
Automatic Differentiation (AD) provides exact derivatives w/o time/effort of deriving and hand-coding them!

- How does AD work? → freshman calculus!
  - Computations are composition of simple operations (+, *, sin(), etc.)
  - Derivatives computed line by line then combined via chain rule.
- Derivatives are as accurate as analytic computation – no finite difference truncation error!
- Great for multi-physics codes (e.g., many Jacobians) and advanced analysis (e.g., sensitivities)
- There are many AD libraries (C++, Fortran, MATLAB, etc.) that can be used (https://en.wikipedia.org/wiki/Automatic_diffentiation) → we use Trilinos package Sacado.

Automatic Differentiation Example:

\[ y = \sin(e^x + x \log x), \quad x = 2 \]

| \( \frac{d}{dx} \) |  
|-------------------|-------------------|-------------------|-------------------|-------------------|-------------------|-------------------|-------------------|-------------------|-------------------|
| \( x \leftarrow 2 \) | \( \frac{dx}{dx} \leftarrow 1 \) | \( t \leftarrow e^x \) | \( \frac{dt}{dx} \leftarrow t \frac{dx}{dx} \) | \( u \leftarrow \log x \) | \( \frac{du}{dx} \leftarrow \frac{1}{x} \frac{dx}{dx} \) | \( v \leftarrow xu \) | \( \frac{dv}{dx} \leftarrow u \frac{dx}{dx} + x \frac{du}{dx} \) | \( w \leftarrow t + v \) | \( \frac{dw}{dx} \leftarrow \frac{dt}{dx} + \frac{dv}{dx} \) | \( y \leftarrow \sin w \) | \( \frac{dy}{dx} \leftarrow \cos(w) \frac{dw}{dx} \) |
| | | | | | | | | | | |  
| | | | | | | | | | |  
| | | | | | | | | | |  
| 1.000 | 7.389 | 0.500 | 1.301 | 8.690 | -1.188 |
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>
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 21\textsuperscript{st} century $\rightarrow$ \textit{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).

Basal boundary $\Gamma_{\beta}$

$\gamma = 12A^{-\frac{1}{2n}} \left( \frac{1}{2} \sum_{ij} \dot{e}_{ij}^2 + \gamma \right)^{\frac{1}{2n} - \frac{1}{2}}$

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.

UQ Workflow
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

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

\[
\Rightarrow \text{Mismatch functional to be minimized:}
\[
m(z) = \frac{1}{2} \left( y^{\text{obs}} - f(z) \right)^T \Gamma_{\text{obs}}^{-1} \left( y^{\text{obs}} - f(z) \right)
\]

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

\[
\text{Covariance of Gaussian posterior related to inverse of misfit Hessian at MAP point**.}
\]

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

Bayesian Inference Workflow

- Dimension reduction via KLE
- Dimension reduction via AS
- Quadratic PCE over active variables
- Laplace posterior at MAP *

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

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

*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

KLE modes = eigenvecs of exponential covariance kernel:

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

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).

* 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:

\[
\pi(\theta|d) = \frac{\pi(d|\theta) \pi(\theta)}{\pi(d)} = \frac{\int \pi(d|\theta) \pi(\theta) \, d\theta}{\pi(d)}
\]
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:

\[
\pi(\theta | d) = \frac{\pi(d | \theta) \pi(\theta)}{\pi(d)} = \frac{\pi(d | \theta) \pi(\theta)}{\int \pi(d | \theta) \pi(\theta) \, d\theta}
\]

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

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

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

\[
\pi(\theta|d) = \frac{\pi(d|\theta) \pi(\theta)}{\pi(d)} = \frac{\int \pi(d|\theta) \pi(\theta) \, d\theta}{\pi(d)}
\]

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

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.
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.
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.
**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 ⇒ *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:* \[ \pi_{\text{post}}(z \mid y^{\text{obs}}) = N(z_{\text{MAP}}, \Gamma_{\text{post}}) \]

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

- **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:
  \[ \Gamma_{\text{post}} = (H_{\text{misfit}}^{\text{PCE}} + \Gamma_{\text{prior}}^{-1})^{-1} \]

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

* Bui-Thanh, Ghattas, Martin, Stadler, SISC, 2013.
Approach 2: Normal Approximation + Low Rank Laplace Approximation

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

- **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:
  \[ \Gamma_{post} = \left( H_{misfit}^{\text{PCE}} + \Gamma_{\text{prior}}^{-1} \right)^{-1} \]

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

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

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

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

- **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:
  \[ \Gamma_{\text{post}} = \left( \mathbf{H}_{\text{misfit}}^{\text{PCE}} + \Gamma_{\text{prior}}^{-1} \right)^{-1} \]
  Dense!

**Symbols:**
- \( V_r, D_r \): eigenvcs, eigenvals of \( \mathbf{H}_{\text{misfit}} \)
- \( \mathbf{H}_{\text{misfit}} = \) prior-preconditioned Hessian of data misfit = \( \Gamma_{\text{prior}}^{-1/2} \mathbf{H}_{\text{misfit}} \Gamma_{\text{prior}}^{1/2} \)
- \( \mathbf{H}_{\text{misfit}} = \) Gauss-Newton portion of Hessian misfit = \( F^\dagger \Gamma_{\text{obs}}^{-1} F \)
- \( \mathbf{V}_r = \Gamma_{\text{prior}}^{-1/2} \mathbf{V}_r, \mathbf{V}_r^\dagger = \) adjoint of \( \mathbf{V}_r \)
- \( \Gamma_{\text{prior}}^{-1} = M^{-1} K, K = \) 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:
  \[
  \Gamma_{\text{post}} = \left(H^{\text{PCE}}_{\text{misfit}} + \Gamma_{\text{prior}}^{-1}\right)^{-1}
  \]
  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 = \(\Gamma_{\text{prior}}^{1/2} H_{\text{misfit}} \Gamma_{\text{prior}}^{1/2}\)
- \(H_{\text{misfit}} = \text{Gauss-Newton portion of Hessian misfit} = F^h \Gamma_{\text{obs}}^{-1} F\)
- \(\tilde{V}_r = \Gamma_{\text{prior}}^{1/2} V_r, \tilde{V}_r^{\diamond} = \text{adjoint of } \tilde{V}_r\)
- \(\Gamma^{-1}_{\text{prior}} = M^{-1} K, K = \text{Laplace stiffness.}\)

---

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:
  \[
  \Gamma_{\text{post}} = \left( H_{\text{misfit}}^{\text{PCE}} + \Gamma_{\text{prior}}^{-1} \right)^{-1}
  \]
  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 \( \tilde{H}_{\text{misfit}} \)
- \( \tilde{H}_{\text{misfit}} = \) 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 = \( F^h \Gamma_{\text{obs}}^{-1} F \)
- \( \tilde{V}_r = \Gamma_{\text{prior}}^{1/2} V_r, \tilde{V}_r^\Diamond = \) 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:**
  \[
  \pi_{\text{pos}}(z \mid y^{\text{obs}}) = N(z_{\text{MAP}}, \Gamma_{\text{post}})
  \]

- **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!

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

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

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

---


Symbols*:

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

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

**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
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](image1.png) ![Mode 2](image2.png) ![Mode 3](image3.png) ![Mode 200](image4.png)
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 *drastically reduces* the *posterior variance*

---

**Mode 1**  **Mode 2**  **Mode 3**  **Mode 200**

**Prior variance**  **Posterior variance**

---

* Sandia National Laboratories
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

$$\beta(u) = \mu_f N \left( \frac{|u|}{|u| + \lambda AN^n} \right)^q \frac{1}{|u|}$$

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

**Figure 2**: gradients can determine directions that significantly impact SLR.

Thickness equation (subglacial hydrology)
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
```
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
**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
  ```

  ```
  computeA_Policy range({0,0,0},{(int)numCells,(int)numNodes,(int)numQPs});
  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:

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

  ```cpp
  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.

  ```cpp
  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**:
  ```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)
  Thread N computes A for (cell,node,qp)=(numCells,numNodes,numQPs)
  
  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.
  ```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
  ```

[Diagram showing single threading with tasks and cores]
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)

- 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
```

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

  ```
  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)

  Kokkos parallelization in ALI is only over **cells**.

- **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.
PISCEES & E3SM Coupling Validation

Sub-shelf melt rates (RRS30to10km resolution)

* Rignot et al., Science, 2013