Ice Sheet System model

Inverse Methods

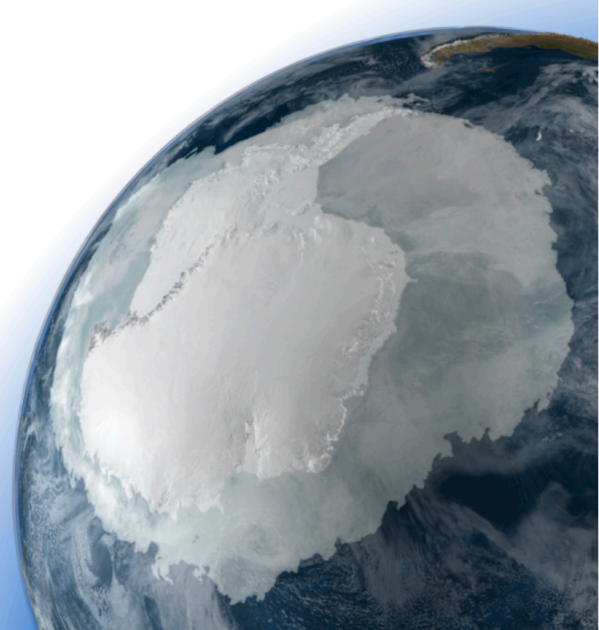
Eric LAROUR[1], Eric RIGNOT[1,3], **Mathieu MORLIGHEM**[1,2], Hélène SEROUSSI[1,2] Chris BORSTAD[1], Feras HABBAL[1,3], Daria HALKIDES[1,4], Behnaz KHAKBAZ[1], John SCHIERMEIER[1], Nicole SCHLEGEL[1]

[1]Jet Propulsion Laboratory - California Institute of Technology

[2]Laboratoire MSSMat, École Centrale Paris, France

[3]University of California, Irvine

[4]Joint Institute for Regional Earth System Science & Engineering, UCLA



Outline

Ice flow model equations

Field equations

1. Stokes flow:

$$\nabla \cdot \boldsymbol{\sigma} + \rho \boldsymbol{g} = \mathbf{0} \tag{1}$$

2. Incompressibility:

$$\nabla \cdot \boldsymbol{v} = 0 \tag{2}$$

3. Constitutive law:

$$\boldsymbol{\sigma}' = \boldsymbol{\sigma} + p\,\mathbf{I} = 2\mu\dot{\boldsymbol{\varepsilon}} \qquad \mu = \frac{B}{2\,\dot{\varepsilon}_e^{1-1/n}} \tag{3}$$

Boundary conditions

Ice/Air interface: Free surface	$\boldsymbol{\sigma} \cdot \boldsymbol{n} = P_{atm}\,\boldsymbol{n} \simeq \mathbf{0}$	on Γ_s
Ice/Ocean interface: water pressure	$\boldsymbol{\sigma} \cdot \boldsymbol{n} = P_w\,\boldsymbol{n}$	on Γ_w
Ice/Bedrock interface (1): lateral friction	$\left(\boldsymbol{\sigma} \cdot \boldsymbol{n} + \alpha^2 \boldsymbol{v}\right)_{\|\|} = \mathbf{0}$	on Γ_b
Ice/Bedrock interface (2): impenetrability	$\boldsymbol{v} \cdot \boldsymbol{n} = -\dot{M}_b\, n_z$	on Γ_b

+ Dirichlet condition

Ice flow model equations

Field equations

1. Stokes flow:

$$\nabla \cdot \boldsymbol{\sigma} + \rho \boldsymbol{g} = \mathbf{0} \tag{1}$$

2. Incompressibility:

$$\nabla \cdot \boldsymbol{v} = 0 \tag{2}$$

3. Constitutive law:

$$\boldsymbol{\sigma}' = \boldsymbol{\sigma} + p\,\mathbf{I} = 2\mu\dot{\boldsymbol{\varepsilon}} \qquad \mu = \frac{B}{2\,\dot{\varepsilon}_e^{1-1/n}} \tag{3}$$

Boundary conditions

Ice/Air interface: Free surface	$\boldsymbol{\sigma} \cdot \boldsymbol{n} = P_{atm}\,\boldsymbol{n} \simeq \mathbf{0}$	on Γ_s
Ice/Ocean interface: water pressure	$\boldsymbol{\sigma} \cdot \boldsymbol{n} = P_w\,\boldsymbol{n}$	on Γ_w
Ice/Bedrock interface (1): lateral friction	$\left(\boldsymbol{\sigma} \cdot \boldsymbol{n} + \alpha^2 \boldsymbol{v}\right)_{\parallel} = \mathbf{0}$	on Γ_b
Ice/Bedrock interface (2): impenetrability	$\boldsymbol{v} \cdot \boldsymbol{n} = -\dot{M}_b\, n_z$	on Γ_b

+ Dirichlet condition

Ice flow model equations

Field equations

1. Stokes flow:

$$\nabla \cdot \boldsymbol{\sigma} + \rho \boldsymbol{g} = \mathbf{0} \tag{1}$$

2. Incompressibility:

$$\nabla \cdot \boldsymbol{v} = 0 \tag{2}$$

3. Constitutive law:

$$\boldsymbol{\sigma}' = \boldsymbol{\sigma} + p\,\mathbf{I} = 2\mu\dot{\boldsymbol{\varepsilon}} \qquad \mu = \frac{B}{2\,\dot{\varepsilon}_e^{1-1/n}} \tag{3}$$

Boundary conditions

Ice/Air interface: Free surface	$\boldsymbol{\sigma} \cdot \boldsymbol{n} = P_{atm}\,\boldsymbol{n} \simeq \mathbf{0}$	on Γ_s
Ice/Ocean interface: water pressure	$\boldsymbol{\sigma} \cdot \boldsymbol{n} = P_w\,\boldsymbol{n}$	on Γ_w
Ice/Bedrock interface (1): lateral friction	$\left(\boldsymbol{\sigma} \cdot \boldsymbol{n} + \alpha^2 \boldsymbol{v}\right)_{\parallel} = \mathbf{0}$	on Γ_b
Ice/Bedrock interface (2): impenetrability	$\boldsymbol{v} \cdot \boldsymbol{n} = -\dot{M}_b\, n_z$	on Γ_b

+ Dirichlet condition

Inverse problems

- Basal friction and ice hardness are difficult to measure
- Use extra datasets to infer unknows

→ ex: surface velocities derived from InSAR

PDE-constrained optimization

Minimize cost function

$$\mathcal{J}(\boldsymbol{v},\alpha) = \frac{1}{2}\int_{\Gamma_s}\left(v_x - v_x^{\text{obs}}\right)^2 + \left(v_y - v_y^{\text{obs}}\right)^2 dS + \mathcal{R}(\alpha) \tag{4}$$

Subject to:

$$\begin{aligned}
\nabla\cdot\mu\left(\nabla\boldsymbol{v}+\nabla\boldsymbol{v}^T\right) - \nabla p + \rho\boldsymbol{g} &= \boldsymbol{0} && \text{in } \Omega \\
\nabla\cdot\boldsymbol{v} &= 0 && \text{in } \Omega \\
\boldsymbol{\sigma}\cdot\boldsymbol{n} &= \boldsymbol{f} && \text{on } \Gamma_s\cup\Gamma_w \\
\left(\boldsymbol{\sigma}\cdot\boldsymbol{n}+\alpha^2\boldsymbol{v}\right)_{||} &= \boldsymbol{0} && \text{on } \Gamma_b \\
\boldsymbol{v}\cdot\boldsymbol{n} &= -\dot{M}_b\, n_z && \text{on } \Gamma_b
\end{aligned} \tag{5}$$

Inverse problems

- Basal friction and ice hardness are difficult to measure
- Use extra datasets to infer unknows

→ ex: surface velocities derived from InSAR

PDE-constrained optimization

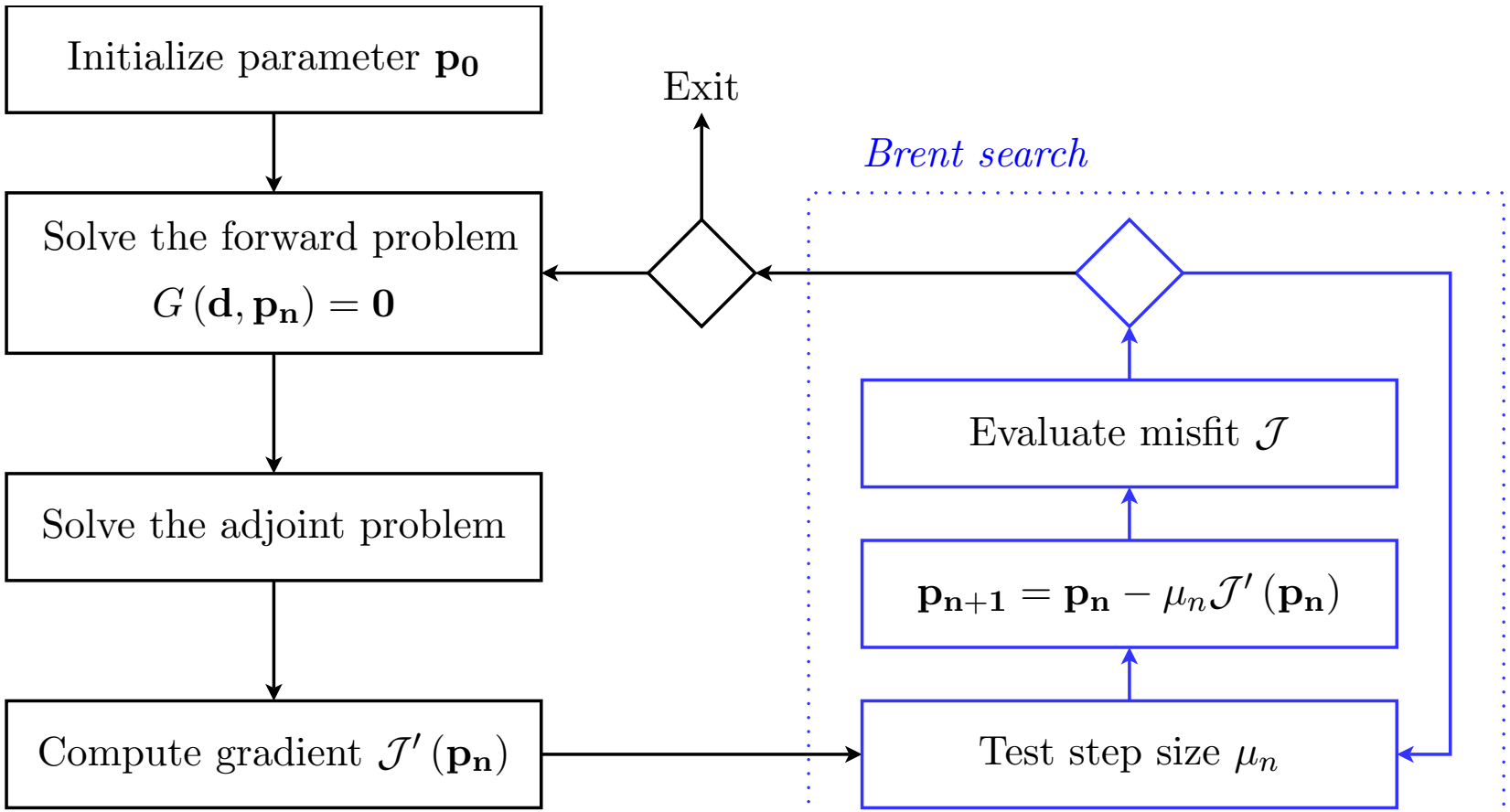
Minimize cost function

$$\mathcal{J}(\boldsymbol{v},\alpha) = \frac{1}{2}\int_{\Gamma_s}\left(v_x - v_x^{\text{obs}}\right)^2 + \left(v_y - v_y^{\text{obs}}\right)^2 dS + \mathcal{R}(\alpha) \tag{4}$$

Subject to:

$$\begin{aligned}
\nabla\cdot\mu\left(\nabla\boldsymbol{v}+\nabla\boldsymbol{v}^T\right) - \nabla p + \rho\boldsymbol{g} &= \boldsymbol{0} && \text{in } \Omega \\
\nabla\cdot\boldsymbol{v} &= 0 && \text{in } \Omega \\
\boldsymbol{\sigma}\cdot\boldsymbol{n} &= \boldsymbol{f} && \text{on } \Gamma_s\cup\Gamma_w \\
\left(\boldsymbol{\sigma}\cdot\boldsymbol{n}+\alpha^2\boldsymbol{v}\right)_{||} &= \boldsymbol{0} && \text{on } \Gamma_b \\
\boldsymbol{v}\cdot\boldsymbol{n} &= -\dot{M}_b\, n_z && \text{on } \Gamma_b
\end{aligned} \tag{5}$$

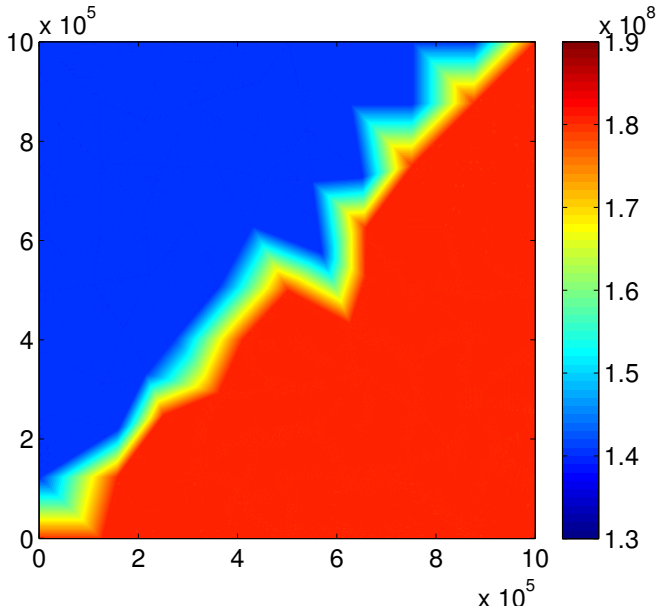
Algorithm of resolution

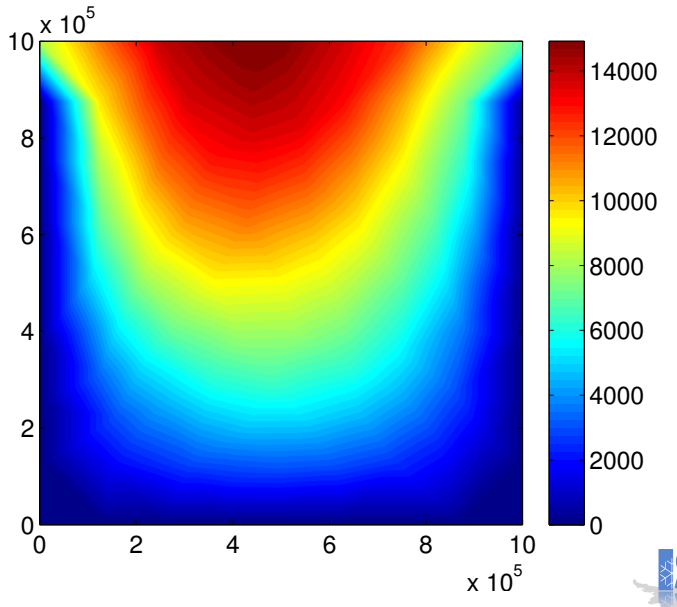


Synthetic case

```
3       %Generate observation
4       md=model;
5       md=triangle(md,'DomainOutline.exp',100000);
6       md=setmask(md,'all','');
7       md=parameterize(md,'Square.par');
8       md=setflowequation(md,'macayeal','all');
9       md.cluster=generic('name',oshostname,'np',2);
10      md=solve(md,DiagnosticSolutionEnum);
```

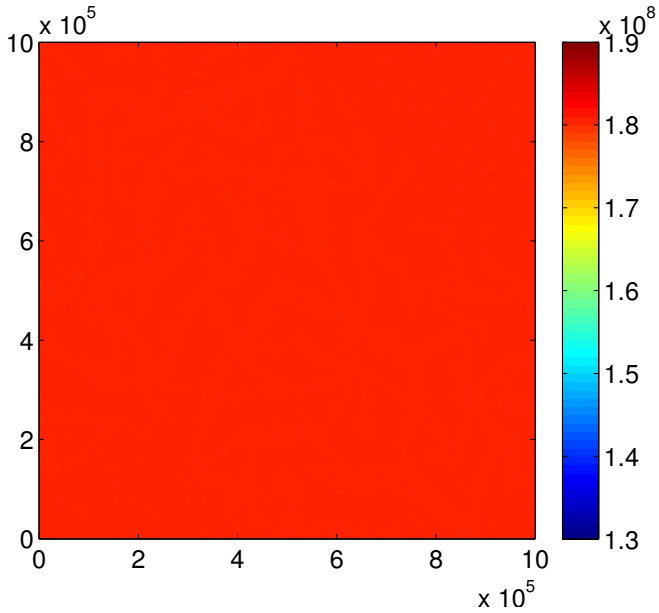
```
25  disp('      creating flow law paramter');
26  md.materials.rheology_B=1.8*10^8*ones(md.mesh.numberofvertices,1);
27  md.materials.rheology_B(find(md.mesh.x<md.mesh.y))=1.4*10^8;
28  md.materials.rheology_n=3*ones(md.mesh.numberofelements,1);
```

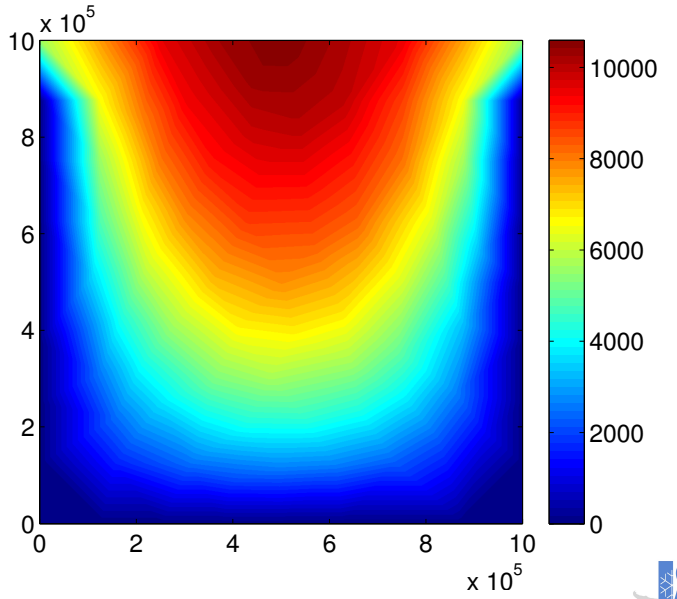




Start from constant hardness

```
16      %Modify rheology, now constant
17      loadmodel('model1.mat');
18      md.materials.rheology_B(:)=1.8*10^8;
19
20      %results of previous run are taken as observations
21      md.inversion.vx_obs=md.results.DiagnosticSolution.Vx;
22      md.inversion.vy_obs=md.results.DiagnosticSolution.Vy;
23      md.inversion.vel_obs=md.results.DiagnosticSolution.Vel;
24
25      md=solve(md,DiagnosticSolutionEnum);
```

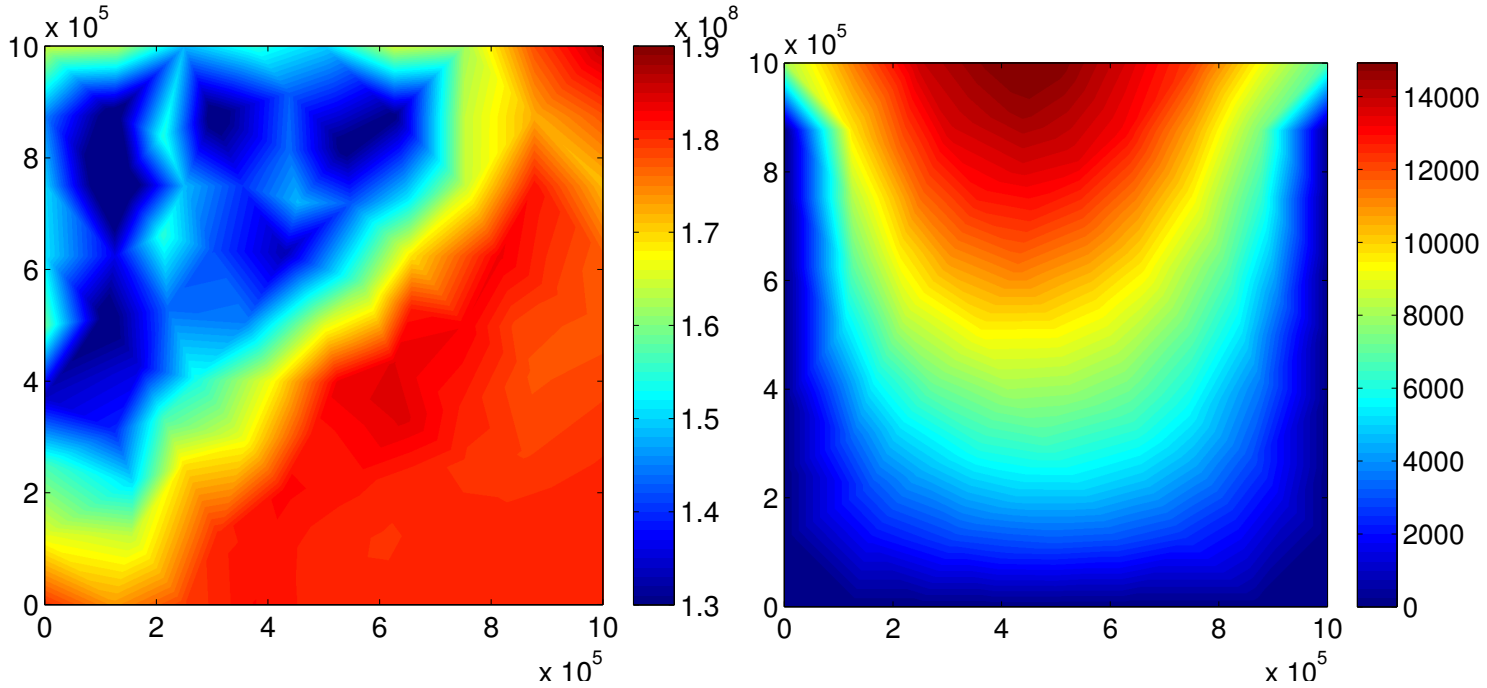




Inverse method

```
31    %invert for rheology
32    loadmodel('model2.mat');
33
34    %Set up inversion parameters
35    nsteps=40;
36    md.inversion.iscontrol=1;
37    md.inversion.control_parameters={'MaterialsRheologyBbar'};
38    md.inversion.nsteps=nsteps;
39    md.inversion.cost_functions=101*ones(nsteps,1);
40    md.inversion.cost_functions_coefficients=ones(md.mesh.numberofvertices,1);
41    md.inversion.maxiter_per_step=10*ones(nsteps,1);
42    md.inversion.step_threshold=.8*ones(nsteps,1);
43    md.inversion.gradient_scaling=10^7*ones(nsteps,1);
44    md.inversion.min_parameters=paterson(273)*ones(md.mesh.numberofvertices,1);
45    md.inversion.max_parameters=paterson(200)*ones(md.mesh.numberofvertices,1);
46
47    %Go solve!
48    md=solve(md,DiagnosticSolutionEnum);
```

Inverse method




Thanks!