ISSM Workshop 2012

Ice Sheet System model

Ice flow models

Chris Borstad[1], Bao Duong[5,1], Feras Habbal[2,1], Daria Halkides[1,3], Michiel Helsen[2], Eric Larour[1], Mathieu Morlighem[2], Lan Nguyen[5,1], Gilberto Pérez[4,1], Eric Rignot[2,1], John Schiermeier[1], Nicole Schlegel[1], **Hélène Seroussi**[1]

[1] Jet Propulsion Laboratory - California Institute of Technology
[2] University of California, Irvine
[3] Joint Institute for Regional Earth System Science & Engineering, UCLA
[4] University of Southern California
[5] Cal Poly Pomona

December 2012

Outline

Ice Sheet flow equations

Incompressibility

$$\forall \boldsymbol{x} \in \Omega \qquad \nabla \cdot \boldsymbol{v} = \mathrm{Tr}\left(\dot{\boldsymbol{\varepsilon}}\right) = \frac{\partial u}{\partial x} + \frac{\partial v}{\partial y} + \frac{\partial w}{\partial z} = 0 \tag{1}$$

- $\boldsymbol{v} = (u, v, w)$ ice velocity (m/yr)
- $\dot{\boldsymbol{\varepsilon}}$ strain rate tensor (yr^{-1})

Incompressible viscous fluid

$$\boldsymbol{\sigma}' = 2\mu\dot{\boldsymbol{\varepsilon}} \tag{2}$$

- $\boldsymbol{\sigma}'$ deviatoric stress
- μ ice viscosity
- $\dot{\boldsymbol{\varepsilon}}$ strain rate tensor

Glen's flow law

$$\mu = \frac{B}{2\,\dot{\varepsilon}_e^{\frac{n-1}{n}}} \tag{3}$$

- B ice hardness
- n Glen's law coefficient ($n = 3$)
- $\dot{\varepsilon}_e$ effective strain rate (second invariant)

Ice Sheet flow equations

Conservation of momentum

$$\forall \boldsymbol{x} \in \Omega \qquad \nabla \cdot \boldsymbol{\sigma}' - \nabla P + \rho \boldsymbol{g} = \boldsymbol{0} \tag{4}$$

Assumptions:

1. Stokes flow (quasi-static assumption)
2. Coriolis effect negligible

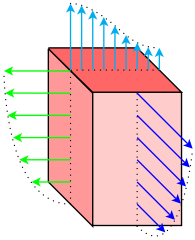
Boundary conditions

Ice/Air interface: Free surface	Γ_s	$\boldsymbol{\sigma} \cdot \boldsymbol{n} = P_{atm}\ \boldsymbol{n} \simeq \boldsymbol{0}$
Ice/Ocean interface: water pressure	Γ_w	$\boldsymbol{\sigma} \cdot \boldsymbol{n} = P_w\ \boldsymbol{n}$
Ice/Bedrock interface (1): lateral friction	Γ_b	$(\boldsymbol{\sigma} \cdot \boldsymbol{n} + \beta \boldsymbol{v})_{\parallel} = \boldsymbol{0}$
Ice/Bedrock interface (2): impenetrability	Γ_b	$\boldsymbol{v} \cdot \boldsymbol{n} = 0$
Side boundaries: Dirichlet	Γ_u	$\boldsymbol{v} = \boldsymbol{v}_{obs}$

Models description

Full-Stokes model:

- Momentum balance + incompressibility
- 3D model
- Four unknowns (v_x, v_y, v_z, p)



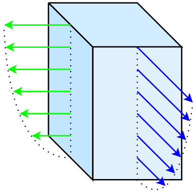
Model equations

$$\left\{\begin{array}{l}
\frac{\partial}{\partial x}\left(2\mu\frac{\partial v_x}{\partial x}\right)+\frac{\partial}{\partial y}\left(\mu\frac{\partial v_x}{\partial y}+\mu\frac{\partial v_y}{\partial x}\right)+\frac{\partial}{\partial z}\left(\mu\frac{\partial v_x}{\partial z}+\mu\frac{\partial v_z}{\partial x}\right)-\frac{\partial p}{\partial x}=0\\
\frac{\partial}{\partial x}\left(\mu\frac{\partial v_x}{\partial y}+\mu\frac{\partial v_y}{\partial x}\right)+\frac{\partial}{\partial y}\left(2\mu\frac{\partial v_y}{\partial y}\right)+\frac{\partial}{\partial z}\left(\mu\frac{\partial v_y}{\partial z}+\mu\frac{\partial v_z}{\partial y}\right)-\frac{\partial p}{\partial y}=0\\
\frac{\partial}{\partial x}\left(\mu\frac{\partial v_x}{\partial z}+\mu\frac{\partial v_z}{\partial x}\right)+\frac{\partial}{\partial y}\left(\mu\frac{\partial v_y}{\partial z}+\mu\frac{\partial v_z}{\partial y}\right)+\frac{\partial}{\partial z}\left(2\mu\frac{\partial v_z}{\partial z}\right)-\frac{\partial p}{\partial z}-\rho g=0
\end{array}\right.$$

$$\frac{\partial v_x}{\partial x}+\frac{\partial v_y}{\partial y}+\frac{\partial v_z}{\partial z}=0$$

Models description

Blatter-Pattyn (BP)



Higher-order model:

- [Blatter, 1995, Pattyn, 2003]
- 3D model
- Horizontal and vertical velocity decoupled
- 2 (v_x, v_y) + 1 (v_z) unknowns

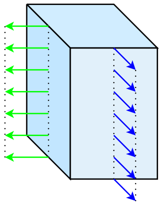
Model equations

$$\left\{\begin{array}{l}
\frac{\partial}{\partial x}\left(2\mu\frac{\partial v_x}{\partial x}\right)+\frac{\partial}{\partial y}\left(\mu\frac{\partial v_x}{\partial y}+\mu\frac{\partial v_y}{\partial x}\right)+\frac{\partial}{\partial z}\left(\mu\frac{\partial v_x}{\partial z}+\mu\frac{\partial v_z}{\partial x}\right)-\frac{\partial p}{\partial x}=0\\
\frac{\partial}{\partial x}\left(\mu\frac{\partial v_x}{\partial y}+\mu\frac{\partial v_y}{\partial x}\right)+\frac{\partial}{\partial y}\left(2\mu\frac{\partial v_y}{\partial y}\right)+\frac{\partial}{\partial z}\left(\mu\frac{\partial v_y}{\partial z}+\mu\frac{\partial v_z}{\partial y}\right)-\frac{\partial p}{\partial y}=0\\
\frac{\partial}{\partial x}\left(\mu\frac{\partial v_x}{\partial z}+\mu\frac{\partial v_z}{\partial x}\right)+\frac{\partial}{\partial y}\left(\mu\frac{\partial v_y}{\partial z}+\mu\frac{\partial v_z}{\partial y}\right)+\frac{\partial}{\partial z}\left(2\mu\frac{\partial v_z}{\partial z}\right)-\frac{\partial p}{\partial z}-\rho g=0
\end{array}\right.$$

$$\frac{\partial v_x}{\partial x}+\frac{\partial v_y}{\partial y}+\frac{\partial v_z}{\partial z}=0$$

Models description

MacAyeal-Morland (SSA)



Shelfy-stream approximation:

- [MacAyeal, 1989]
- 2D model
- Horizontal and vertical velocity decoupled
- 2 (v_x, v_y) + 1 (v_z) unknowns

Model equations

$$\begin{cases} \frac{\partial}{\partial x}\left(2\mu\frac{\partial v_x}{\partial x}\right) + \frac{\partial}{\partial y}\left(\mu\frac{\partial v_x}{\partial y} + \mu\frac{\partial v_y}{\partial x}\right) + \frac{\partial}{\partial z}\left(\mu\frac{\partial v_x}{\partial z} + \mu\frac{\partial v_z}{\partial x}\right) - \frac{\partial p}{\partial x} = 0 \\ \frac{\partial}{\partial x}\left(\mu\frac{\partial v_x}{\partial y} + \mu\frac{\partial v_y}{\partial x}\right) + \frac{\partial}{\partial y}\left(2\mu\frac{\partial v_y}{\partial y}\right) + \frac{\partial}{\partial z}\left(\mu\frac{\partial v_y}{\partial z} + \mu\frac{\partial v_z}{\partial y}\right) - \frac{\partial p}{\partial y} = 0 \\ \frac{\partial}{\partial x}\left(\mu\frac{\partial v_x}{\partial z} + \mu\frac{\partial v_z}{\partial x}\right) + \frac{\partial}{\partial y}\left(\mu\frac{\partial v_y}{\partial z} + \mu\frac{\partial v_z}{\partial y}\right) + \frac{\partial}{\partial z}\left(2\mu\frac{\partial v_z}{\partial z}\right) - \frac{\partial p}{\partial z} - \rho g = 0 \end{cases}$$

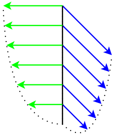
$$\frac{\partial v_x}{\partial x} + \frac{\partial v_y}{\partial y} + \frac{\partial v_z}{\partial z} = 0$$

Models description

Shallow ice approximation:

- [Hutter, 1983]
- 3D analytical model
- 2 unknowns (v_x, v_y) computed separately

Hutter (SIA)



(v_x, v_y)

Model equations

$$\begin{cases} \frac{\partial}{\partial x}\left(2\mu\frac{\partial v_x}{\partial x}\right) + \frac{\partial}{\partial y}\left(\mu\frac{\partial v_x}{\partial y} + \mu\frac{\partial v_y}{\partial x}\right) + \frac{\partial}{\partial z}\left(\mu\frac{\partial v_x}{\partial z} + \mu\frac{\partial v_z}{\partial x}\right) - \frac{\partial p}{\partial x} = 0 \\ \frac{\partial}{\partial x}\left(\mu\frac{\partial v_x}{\partial y} + \mu\frac{\partial v_y}{\partial x}\right) + \frac{\partial}{\partial y}\left(2\mu\frac{\partial v_y}{\partial y}\right) + \frac{\partial}{\partial z}\left(\mu\frac{\partial v_y}{\partial z} + \mu\frac{\partial v_z}{\partial y}\right) - \frac{\partial p}{\partial y} = 0 \\ \frac{\partial}{\partial x}\left(\mu\frac{\partial v_x}{\partial z} + \mu\frac{\partial v_z}{\partial x}\right) + \frac{\partial}{\partial y}\left(\mu\frac{\partial v_y}{\partial z} + \mu\frac{\partial v_z}{\partial y}\right) + \frac{\partial}{\partial z}\left(2\mu\frac{\partial v_z}{\partial z}\right) - \frac{\partial p}{\partial z} - \rho g = 0 \end{cases}$$

$$\frac{\partial v_x}{\partial x} + \frac{\partial v_y}{\partial y} + \frac{\partial v_z}{\partial z} = 0$$

Material non-linearity

Model equations

$$\begin{cases} \frac{\partial}{\partial x}\left(2\mu\frac{\partial v_x}{\partial x}\right)+\frac{\partial}{\partial y}\left(\mu\frac{\partial v_x}{\partial y}+\mu\frac{\partial v_y}{\partial x}\right)+\frac{\partial}{\partial z}\left(\mu\frac{\partial v_x}{\partial z}+\mu\frac{\partial v_z}{\partial x}\right)-\frac{\partial p}{\partial x}=0 \\ \frac{\partial}{\partial x}\left(\mu\frac{\partial v_x}{\partial y}+\mu\frac{\partial v_y}{\partial x}\right)+\frac{\partial}{\partial y}\left(2\mu\frac{\partial v_y}{\partial y}\right)+\frac{\partial}{\partial z}\left(\mu\frac{\partial v_y}{\partial z}+\mu\frac{\partial v_z}{\partial y}\right)-\frac{\partial p}{\partial y}=0 \\ \frac{\partial}{\partial x}\left(\mu\frac{\partial v_x}{\partial z}+\mu\frac{\partial v_z}{\partial x}\right)+\frac{\partial}{\partial y}\left(\mu\frac{\partial v_y}{\partial z}+\mu\frac{\partial v_z}{\partial y}\right)+\frac{\partial}{\partial z}\left(2\mu\frac{\partial v_z}{\partial z}\right)-\frac{\partial p}{\partial z}-\rho g=0 \end{cases}$$

$$\frac{\partial v_x}{\partial x}+\frac{\partial v_y}{\partial y}+\frac{\partial v_z}{\partial z}=0$$

Glen's flow law

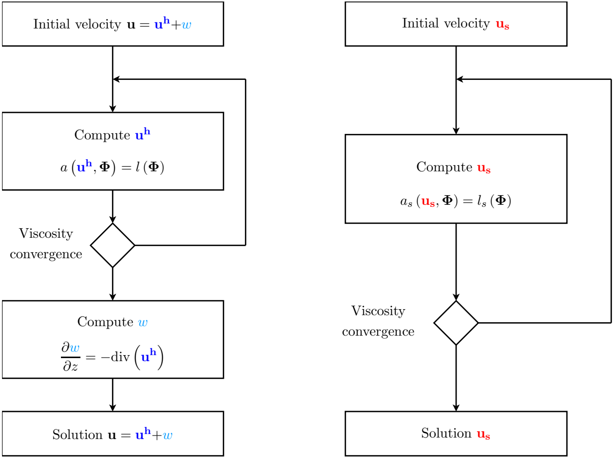
$$\mu=\frac{B}{2\,\dot{\varepsilon}_e^{\frac{n-1}{n}}} \tag{5}$$

- B ice hardness
- n Glen's law coefficient ($n=3$)
- $\dot{\varepsilon}_e$ effective strain rate (second invariant)

$\rightarrow$ Treatment of non-linearity with fixed point

Material non-linearity

Treatment of non-linearity with fixed point:



Vertical velocity computed with incompressibility for 2d shelfy-stream and 3d Blatter/Pattyn modes.

Flow equation

`setflowequation` is used to generate the approximation used to compute the velocity

- Arguments:
  1. model
  2. approximation names
  3. approximation domains
- Domains can be Argus files or array of element flags
- Approximation available
  - stokes (Full-Stokes model)
  - pattyn (Higher-order model)
  - macayeal (Shallow Shelf Approximation)
  - hutter (Shallow Ice Approximation)
- Possibility of coupling models

Flow equation

`setflowequation` is used to generate the approximation used to compute the velocity

- Examples

```
md=setflowequation(md,'hutter','all')
md=setflowequation(md,'stokes','all')
md=setflowequation(md,'macayeal','all')
md=setflowequation(md,'pattyn','all')
```

- To diplay the type of approximation:

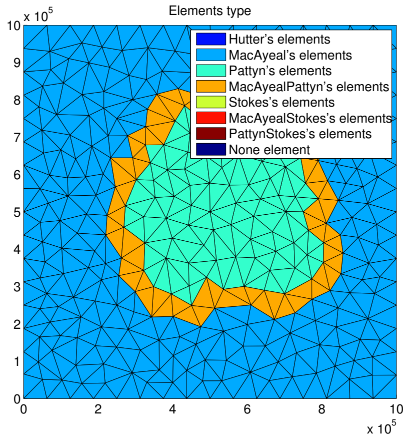
```
>> plotmodel(md,'data','elements_type')
```

JPL ISSM

Flow equation

- To diplay the type of approximation:

```
1  >> plotmodel(md,'data','elements_type')
```



Flow equation class

```
>> md.flowequation

ans =

   flow equation parameters:
            ismacayealpattyn       : 0    -- is the macayeal or pattyn approximation used ?
            ishutter               : 0    -- is the shallow ice approximation used ?
            isstokes               : 0    -- are the Full-Stokes equations used ?
            vertex_equation        : N/A  -- flow equation for each vertex
            element_equation       : N/A  -- flow equation for each element
            bordermacayeal         : N/A  -- vertices on MacAyeal's border (for tiling)
            borderpattyn           : N/A  -- vertices on Pattyn's border (for tiling)
            borderstokes           : N/A  -- vertices on Stokes' border (for tiling)
```

Diagnostic class

```
>> md.diagnostic

ans =

   Diagnostic solution parameters:

     Convergence criteria:
        restol                   : 0.0001          -- mechanical equilibrium residue convergence criterion
        reltol                   : 0.01            -- velocity relative convergence criterion, NaN -> not applied
        abstol                   : 10              -- velocity absolute convergence criterion, NaN -> not applied
        maxiter                  : 100             -- maximum number of nonlinear iterations
        viscosity_overshoot      : 0               -- over-shooting constant new=new+C*(new-old)

     boundary conditions:
        spcvx                    : N/A             -- x-axis velocity constraint (NaN means no constraint)
        spcvy                    : N/A             -- y-axis velocity constraint (NaN means no constraint)
        spcvz                    : N/A             -- z-axis velocity constraint (NaN means no constraint)
        icefront                 : N/A             -- segments on ice front list (last column 0-> Air, 1-> Water, ...
                2->Ice

     Rift options:
        rift_penalty_threshold : 0                 -- threshold for instability of mechanical constraints
        rift_penalty_lock        : 10              -- number of iterations before rift penalties are locked

     Penalty options:
        penalty_factor           : 3               -- offset used by penalties: penalty = Kmax*10^offset
        vertex_pairing           : N/A             -- pairs of vertices that are penalized

     Other:
        shelf_dampening          : 0               -- use dampening for floating ice ?  Only for Stokes model
        stokesreconditioning     : 10000000000000 -- multiplier for incompressibility equation.  Only for Stokes model
        referential              : N/A             -- local referential
        requested_outputs        : N/A             -- additional outputs requested
```

Boundary conditions

Boundary conditions created automatically or manually

- Automatically:

```
>> md=SetIceSheetBC(md)
>> md=SetIceShelfBC(md,'Front.exp')
>> md=SetMarineIceSheefBC(md,'Front.exp')
```

- Manually: fields to change
  - `md.diagnostic.spcvx`
  - `md.diagnostic.spcvy`
  - `md.diagnostic.spcvz`
  - `md.diagnostic.icefront`

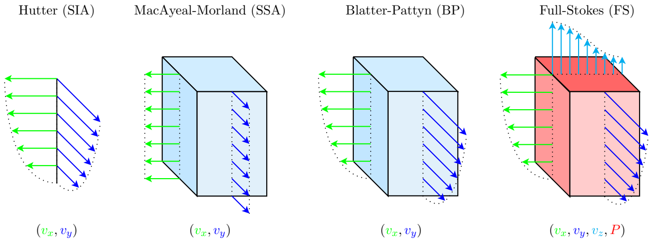
- To diplay the boundary conditions

```
>> plotmodel(md,'data','BC')
```

Models description

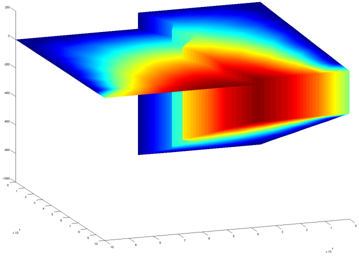
"Everything should be made as simple as possible, but no simpler." Albert Einstein

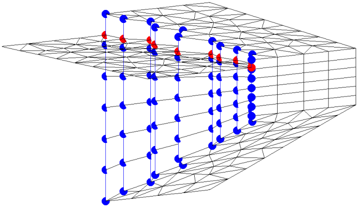
Model	Dim.	Unknowns	Reference
Full-Stokes (FS)	3d	4	[Stokes, 1845]
Blatter-Pattyn (BP)	3d	2 + 1	[Blatter, 1995, Pattyn, 2003]
Shallow shelf (SSA)	2d	2 + 1	[MacAyeal, 1989]
Shallow ice (SIA)	2d	2 + 1	[Hutter, 1983]



Penalty method

- Only to couple SSA and HO
- Very stiff spring to penalize differences between degrees of freedom





Using penalties to couple models:

```
md=setflowequation(md,'macayeal','FloatingIce.exp','fill','pattyn','coupling','penalties')
```

Ice flow models

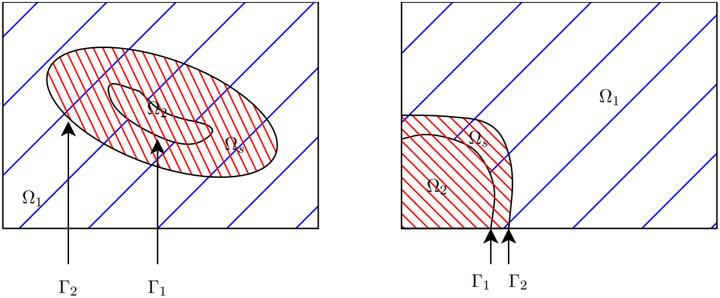
Ice flow equations
Approximations implemented
Ice flow equation
Diagnostic parameters
Boundary conditions

Combining models
Methods implemented in ISSM
Penalties
Tiling method
Usage

EISMINT

Domain Decomposition

- $\Omega = \Omega_1 \cup \Omega_2$
- $\Omega_S = \Omega_1 \cap \Omega_2 \neq \varnothing$
- $\boldsymbol{u} = \boldsymbol{u_1}\,|_{\Omega_1} + \boldsymbol{u_2}\,|_{\Omega_2} \in \tilde{V}(\Omega) = (V_1(\Omega_1) + V_2(\Omega_2))$

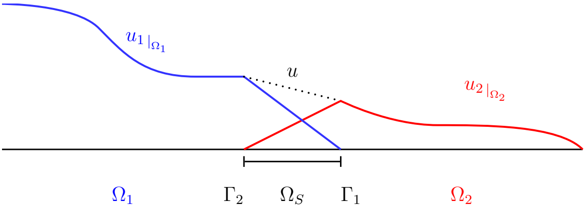


$$\text{Find } \boldsymbol{u} = \boldsymbol{u_1}\,|_{\Omega_1} + \boldsymbol{u_2}\,|_{\Omega_2} \in \tilde{V},$$

$$\forall (\boldsymbol{v_1}, \boldsymbol{v_2}) \in \tilde{V} \qquad a(\boldsymbol{u_1} + \boldsymbol{u_2}, \boldsymbol{v_1} + \boldsymbol{v_2}) = l(\boldsymbol{v_1} + \boldsymbol{v_2})$$

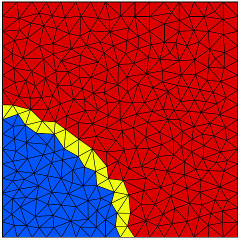
→ Infinite number of solutions for the continuous problem

Discretization



We take advantage of the discretization to avoid the redundancy:

→ Create one layer of elements in the superposition zone



Multi-model formulation

Two different models: a_1, a_2 and l_1, l_2

Find $\boldsymbol{u} = \boldsymbol{u_1}\,|_{\Omega_1} + \boldsymbol{u_2}\,|_{\Omega_2} \in (V_1 + V_2)$, such that:

$$\forall \boldsymbol{v} = \boldsymbol{v_1}\,|_{\Omega_1} + \boldsymbol{v_2}\,|_{\Omega_2} \in (V_1 + V_2)$$

$$\underbrace{a_1\left(\boldsymbol{u_1}\,|_{\Omega_1}, \boldsymbol{v_1}\,|_{\Omega_1}\right)}_{\text{model 1}} + \underbrace{a_2\left(\boldsymbol{u_2}\,|_{\Omega_2}, \boldsymbol{v_2}\,|_{\Omega_2}\right)}_{\text{model 2}} + \underbrace{a_2\left(\boldsymbol{u_1}\,|_{\Omega_1}, \boldsymbol{v_2}\,|_{\Omega_2}\right) + a_1\left(\boldsymbol{u_2}\,|_{\Omega_2}, \boldsymbol{v_1}\,|_{\Omega_1}\right)}_{\text{model coupling}} = \underbrace{l_1\left(\boldsymbol{v_1}\,|_{\Omega_1}\right)}_{\text{model 1}} + \underbrace{l_2\left(\boldsymbol{v_2}\,|_{\Omega_2}\right)}_{\text{model 2}}$$

- Coupling different mechanical models
- Easy to implement (local modification of stiffness matrices)

Flow equation

`setflowequation` is used to generate the approximation used to compute the velocity

- Examples

```
md=setflowequation(md,'pattyn',md.elementongroundedice,'fill','macayeal','coupling','penalties')
md=setflowequation(md,'pattyn',md.elementongroundedice,'fill','macayeal','coupling','tiling')
md=setflowequation(md,'stokes','Contour.exp','fill','pattyn')
```

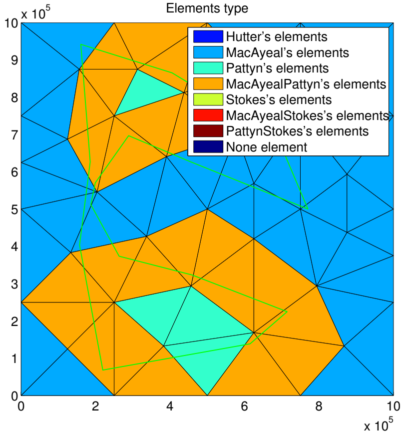
- Use `exptool` to create EXP contours

```
>> exptool('Contour.exp')
```

Flow equation

- To diplay the type of approximation:

```
plotmodel(md,'data','elements_type','edgecolor','k','expdisp','Contour.exp')
```

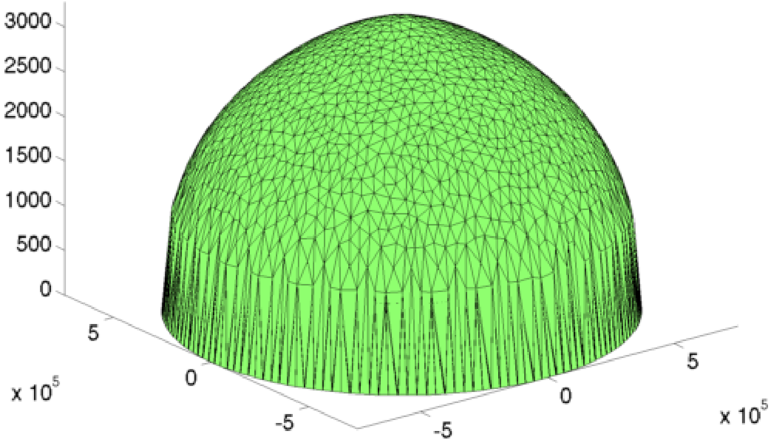


EISMINT

- European Ice Sheet Modeling INiTiative

Objectives:

- Test and compare existing numerical ice-sheet, ice-shelf, and glacier models



EISMINT

- Create mesh using `roundmesh`
- Circle of radius 750 km

EISMINT

- Create mesh using `roundmesh`
- Circle of radius 750 km

Solution:

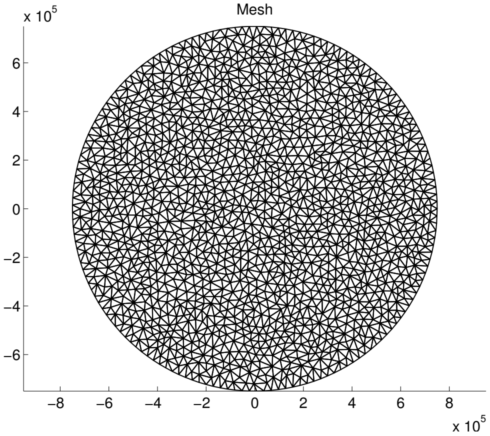
- `md=roundmesh(md,750000,30000);`

EISMINT

- Create mesh using `roundmesh`
- Circle of radius 750 km

Solution:

- `md=roundmesh(md,750000,30000);`
- `plotmodel(md,'data','mesh');`



EISMINT

- Define mask
- All grounded ice

EISMINT

- Define mask
- All grounded ice

Solution:

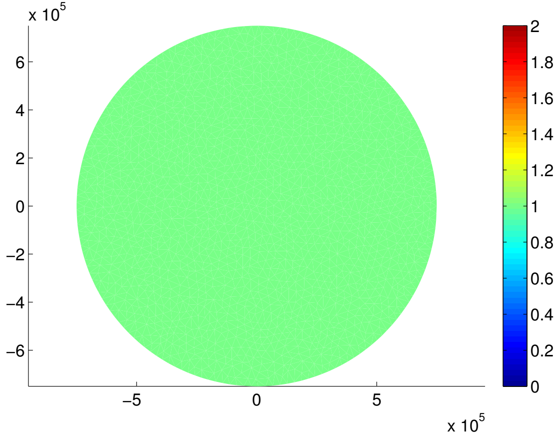
- `md=setmask(md,'','');`

EISMINT

- Define mask
- All grounded ice

Solution:

- `md=setmask(md,'','');`
- `plotmodel(md,'data',md.mask.elementongroundedice);`



EISMINT

- Parameterize
- use `EISMINT.par`

EISMINT

- Parameterize
- use `EISMINT.par`

Solution:

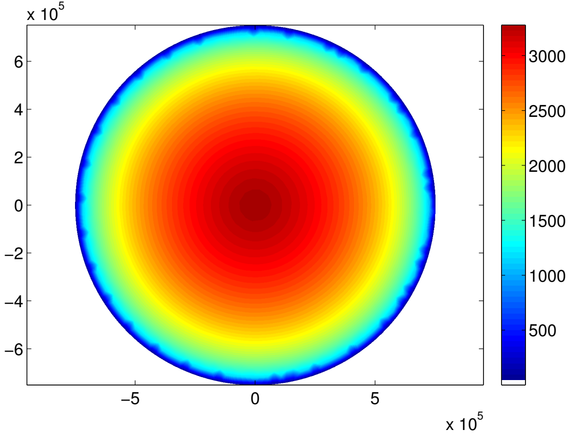
- `md=parameterize(md,'EISMINT.par');`

EISMINT

- Parameterize
- use `EISMINT.par`

Solution:

- `md=parameterize(md,'EISMINT.par');`
- `plotmodel(md,'data',md.geometry.surface)`



EISMINT

- Extrude to create a 3d model
- use 10 layers equally distributed

EISMINT

- Extrude to create a 3d model
- use 10 layers equally distributed

Solution:

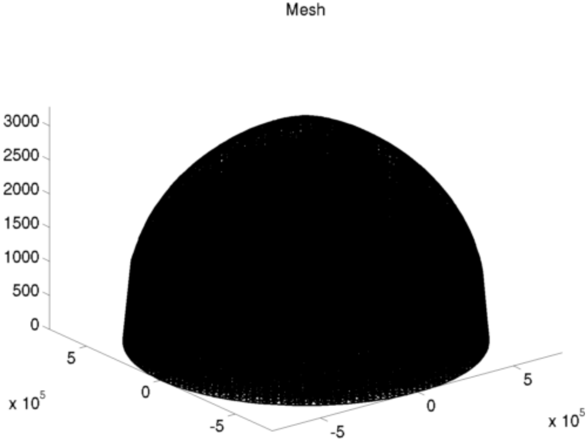
- `md=extrude(md,10,1);`

EISMINT

- Extrude to create a 3d model
- use 10 layers equally distributed

Solution:

- `md=extrude(md,10,1);`
- `plotmodel(md,'data','mesh')`



EISMINT

- Set ice flow model
- Start with SIA (hutter), then HO (pattyn)

EISMINT

- Set ice flow model
- Start with SIA (hutter), then HO (pattyn)

Solution:

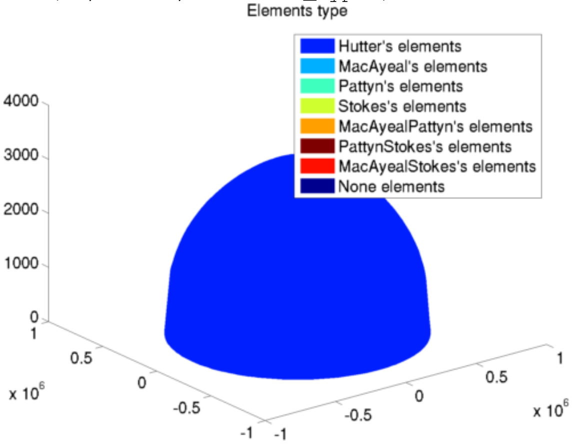
- `md=setflowequation(md,'hutter','all');`

EISMINT

- Set ice flow model
- Start with SIA (hutter), then HO (pattyn)

Solution:

- `md=setflowequation(md,'hutter','all');`
- `plotmodel(md,'data','elements_type')`



EISMINT

- Adjust boundary conditions
- No sliding at the bed

EISMINT

- Adjust boundary conditions
- No sliding at the bed

Solution:

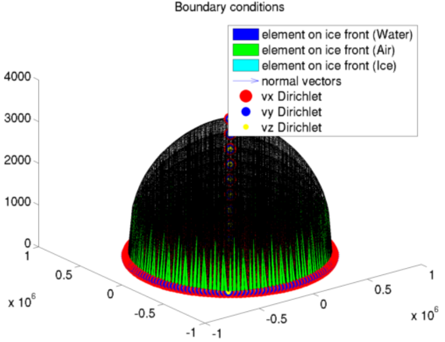
- `pos=find(md.mesh.vertexonbed);`
- `md.diagnostic.spcvx(pos)=0;`
- `md.diagnostic.spcvy(pos)=0;`
- `md.diagnostic.spcvz(pos)=0;`

EISMINT

- Adjust boundary conditions
- No sliding at the bed

Solution:

- `pos=find(md.mesh.vertexonbed);`
- `md.diagnostic.spcvx(pos)=0;`
- `md.diagnostic.spcvy(pos)=0;`
- `md.diagnostic.spcvz(pos)=0;`
- `plotmodel(md,'data','BC')`



Bibliography I

Blatter, H. (1995).
Velocity and stress-fields in grounded glaciers: A simple algorithm for including deviatoric stress gradients.
J. Glaciol., 41(138):333–344.

Hutter, K. (1983).
Theoretical glaciology: material science of ice and the mechanics of glaciers and ice sheets.
D. Reidel Publishing Co, Dordrecht, The Netherlands.

MacAyeal, D. (1989).
Large-scale ice flow over a viscous basal sediment: Theory and application to Ice Stream B, Antarctica.
J. Geophys. Res., 94(B4):4071–4087.

Pattyn, F. (2003).
A new three-dimensional higher-order thermomechanical ice sheet model: Basic sensitivity, ice stream development, and ice flow across subglacial lakes.
J. Geophys. Res., 108(B8):1–15.

Stokes, G. (1845).
On the theories of internal friction of fluids in motion.
Trans. Cambridge Philos. Soc., 8:287–305.


Thanks!