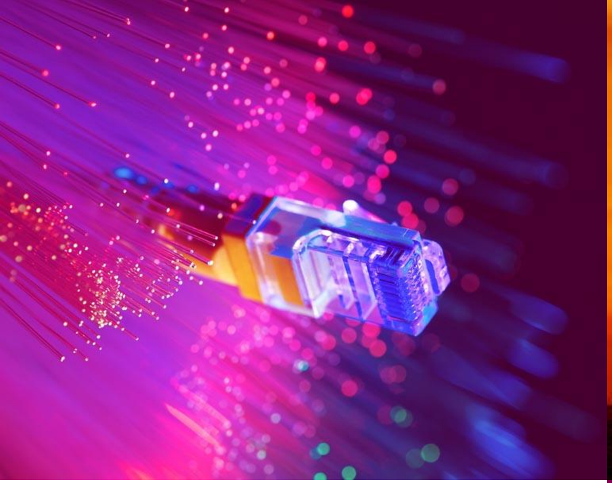


CSC

Elmer/Ice advanced workshop 2017

Grenoble, France


CSC – Finnish research, education, culture and public administration ICT knowledge center

Calving Models in Elmer/Ice

Thomas Zwinger, Dorothée Vallot,
Shahbaz Memnon, Joe Todd, Jan Åström

Continuum damage models (CDM)

Continuum damage model

- Existing CDM in Elmer/Ice (where is actually the code/example?)
    - Krug, J., J. Weiss, O. Gagliardini and G. Durand, 2014. ***Combining damage and fracture mechanics to model calving***, The Cryosphere, **8**, 2101-2117, doi:10.5194/tc-8-2101-2014.
- Main difficulty (to my understanding) is damage transport
- Changing rheology with respect to damage parameter

Calving using in-situ stress-based criterions
(Joe Todd's stuff)

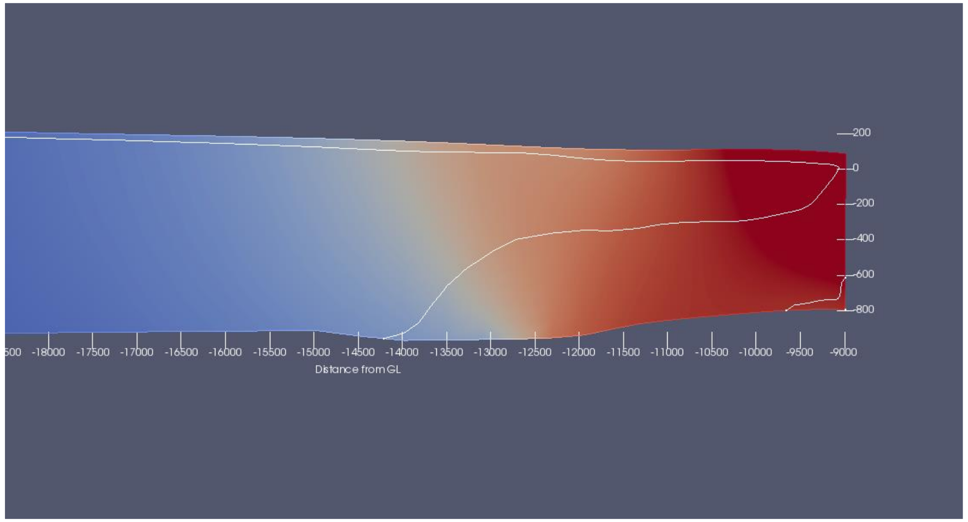
Calving using maximum extensional stress

- Code contributed by **Joe Todd** (Scott Polar/St. Andrews)
- Uses 3D Nye criterion to determine place of failure
  - Determining the max. principal stress, σ_3, using ComputeEigenValues
  - Checking for places with $\sigma_3 > \sigma_{thresh.} \sim 0$
  - In-situ calving criterion (in opposite to CDM + transport)
- The "beef" is the calving/remeshing implementation

Elmer/Ice Calving Models

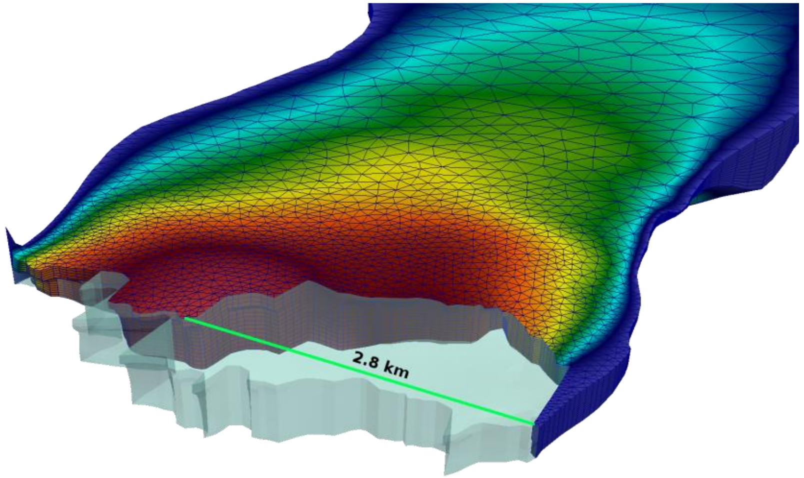
2D Calving:

- Calving = point on line
- Manipulate original mesh (accordion)
- Simple, fast, serial



3D Calving:

- Calving = line on surface
- Complete remeshing
- Complex, expensive, parallel
- Better



Elmer/Ice Calving Solvers

2D Calving:

- Calving.F90
- TwoMeshes.F90

2,000 lines of code

Both use the 'crevasse depth calving criterion' but others could be implemented easily.

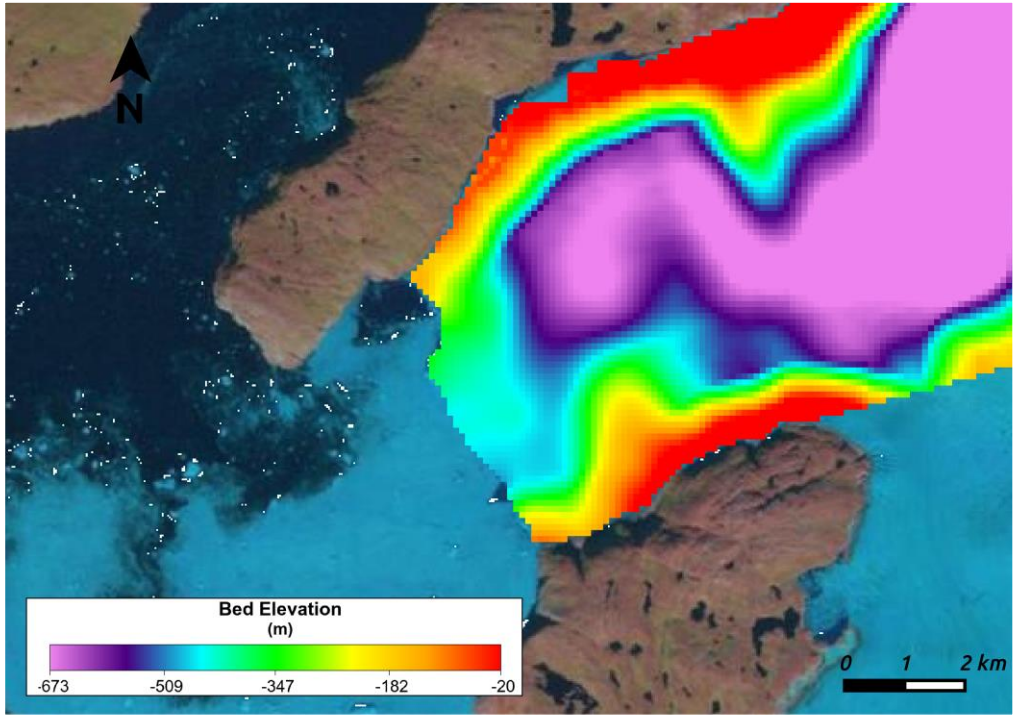
3D Calving:

- Calving3D.F90
- CalvingRemesh.F90
- ProjectCalving.F90
- CalvingGeometry.F90
- ComputeCalvingNormal.F90
- CalvingFrontAdvance3D.F90

11,000 lines of code

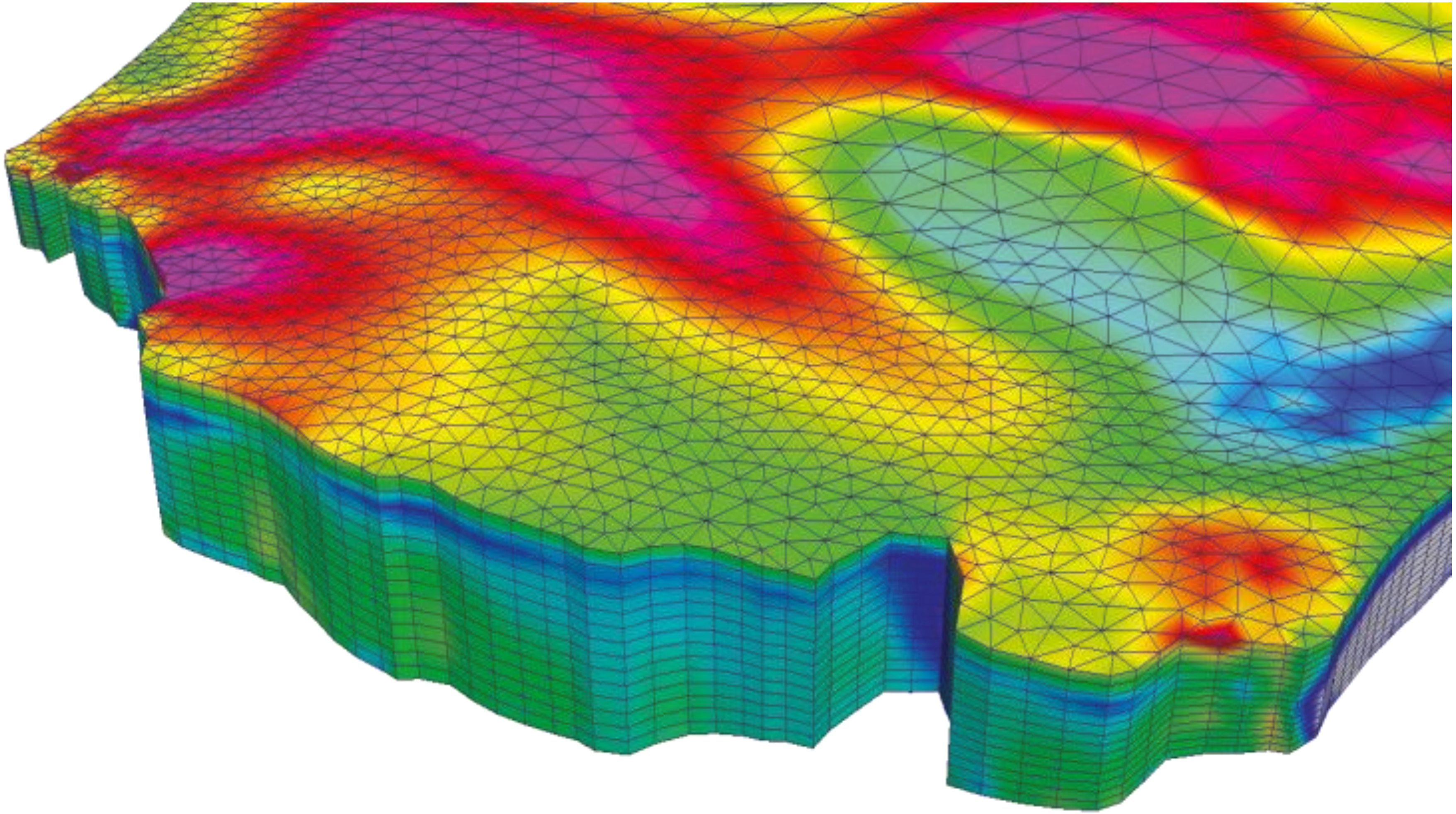
Dependencies

- **Software:**
- GMSH for remeshing
- NETCDF for GridDataReader
- Linux?
- **Data:**
- Accurate bed topography
- Initial terminus position
- Velocity for inversions

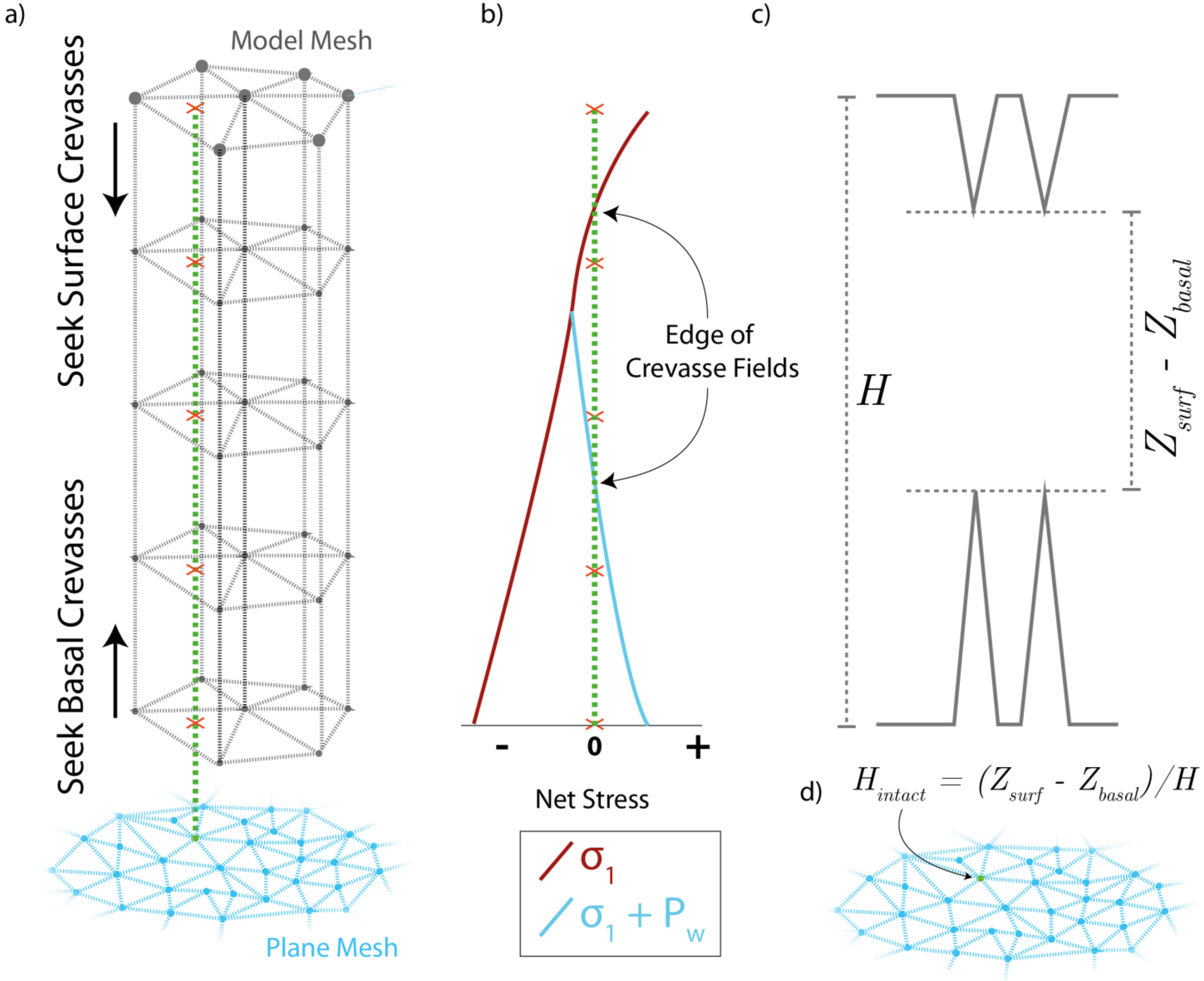


Basal topography produced via mass conservation.

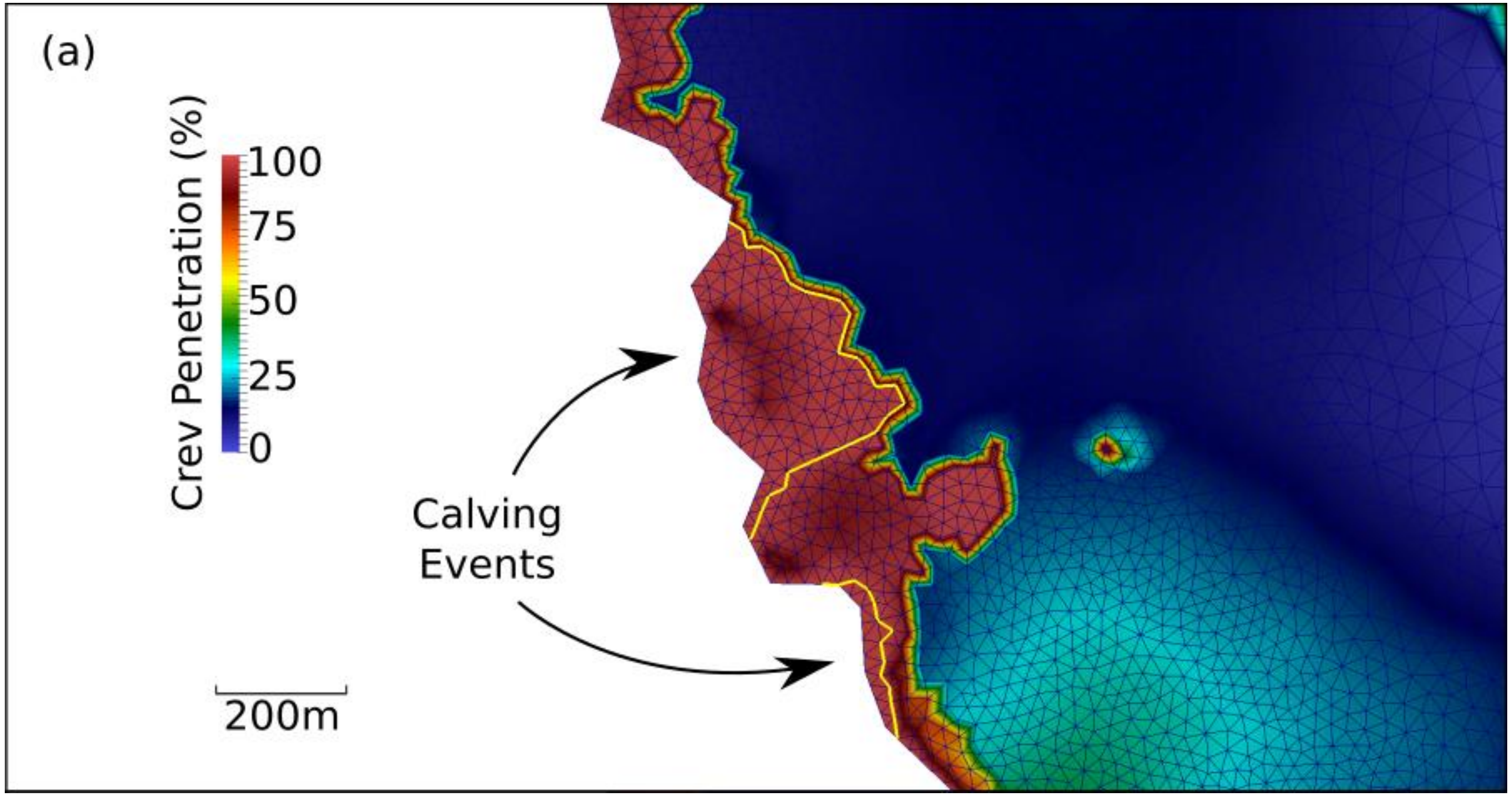
Predicting Calving



Predicting Calving



Predicting Calving



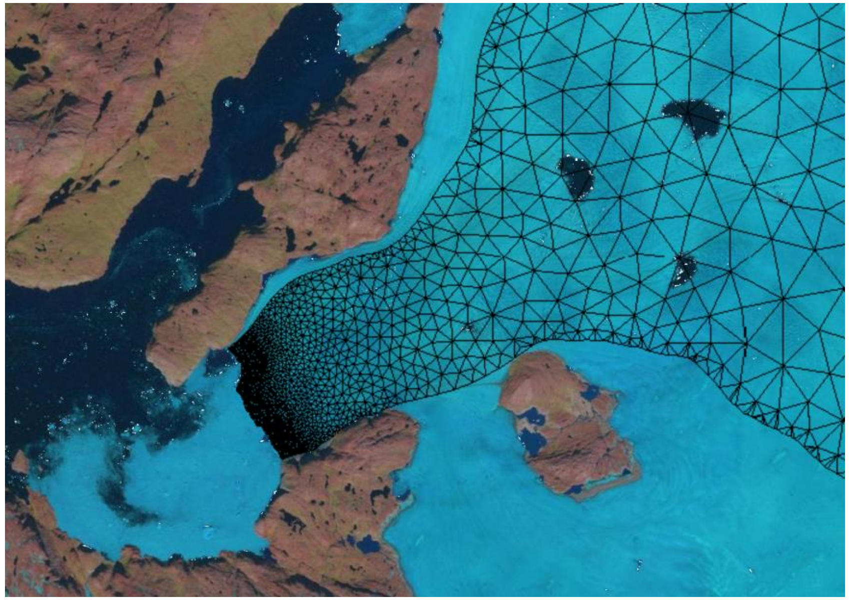
Remeshing

Input: Calving vector defined on front

Output: Good quality mesh with post-calving geometry & all field variables.

Method:

1. Produces 'post-calving' footprint
2. Mesh it in GMSH
3. Extrude it
4. Deform it
5. Interpolate variables



Terminus Advance

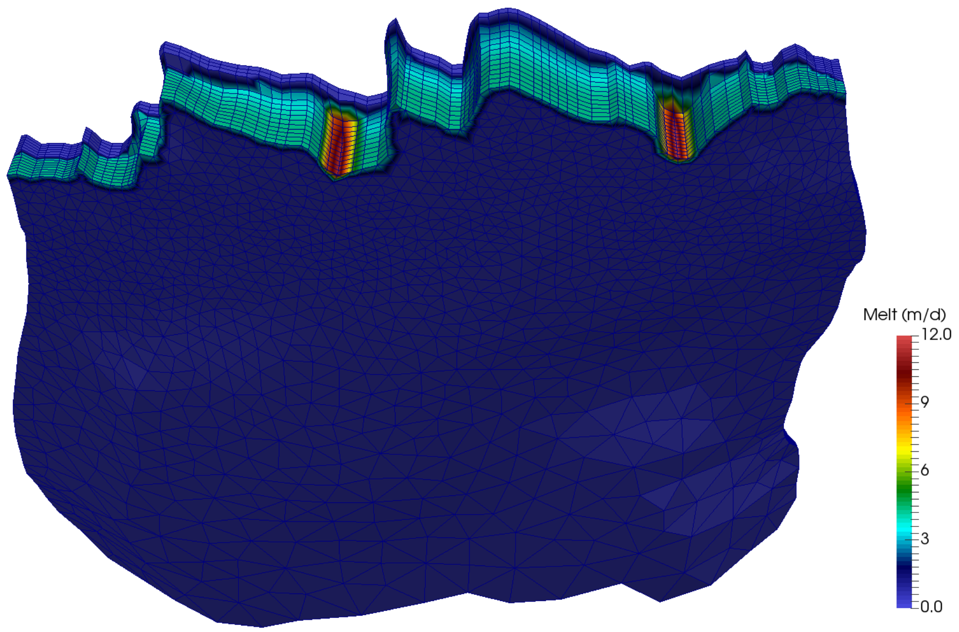
Continuous process, unlike calving

FreeSurfaceSolver doesn't work

CalvingFrontAdvance.F90 computes:

$$\vec{d} = (\vec{u} - a_{\perp}\vec{n})dt$$

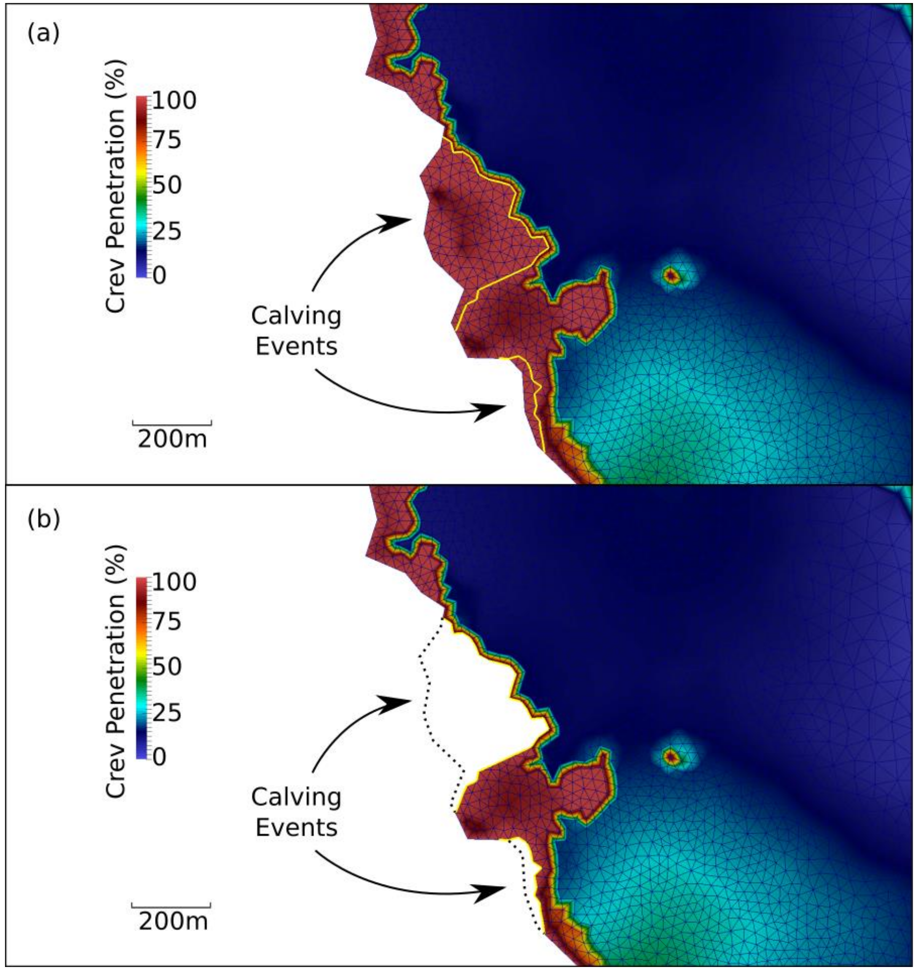
So nodes are free to move in any direction.



Adaptive Timestepping

- **Problem:** Calving events trigger 'follow-up' events, but timestepping introduces artificial delay.

- **Solution:** If a large calving event occurs, change the timestep size to quasi-steady state (1 day => 1 second) and recompute velocity, stress, calving.



Robustness & Stability

- Unsupervised remeshing causes issues
- “Check NS” looks for *suspicious* velocity solution and remeshes/rewinds
- Looks for:
  1. Convergence failure
  2. Very high velocity
  3. Large *changes* in velocity

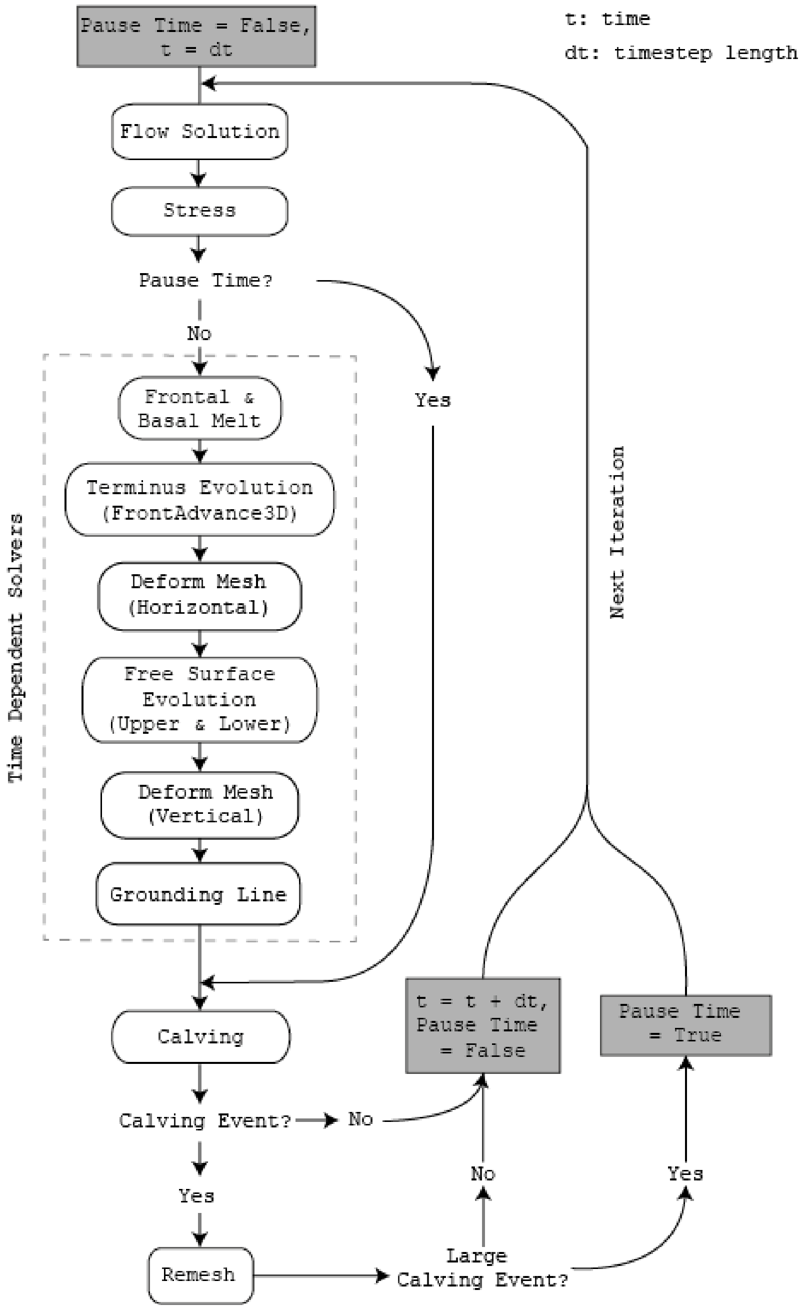
```
Solver 5
  Equation = String "Check NS"
  Procedure = File "ElmerIceSolvers" "CheckFlowConvergence"

  Flow Solver Name = String "Flow Solution"
  Maximum Flow Solution Divergence = Real 1.3
  Maximum Velocity Magnitude = Real 1.0E6
  First Time Max Expected Velocity = Real 8.0E4

  !list of solvers to skip this time if NS fails to converge
  Switch Off Equation 1 = String "StressSolver"
  Switch Off Equation 2 = String "3D Calving"
  Switch Off Equation 3 = String "Free Surface Top"
  Switch Off Equation 4 = String "Free Surface Bottom"
  Switch Off Equation 5 = String "Front Advance"
  Switch Off Equation 6 = String "Longitudinal Mesh Update"
  Switch Off Equation 7 = String "Vertical Mesh Update"
End
```

Typical Simulation

- Compute velocity & stress (and check!)
- Advance front
- Evolve upper & lower surfaces
- Look for calving
- Remesh, interpolate & continue



Getting Help

- Look at the **test cases** in: elmerice/Tests/Calving*
- Look at the **Elmer/Ice wiki** –> Problems -> Calving
- Read the **source code**!

```
tElement % nodeindexes) .GT. 0))) CYCLE
BC elements, stupid way of doing it, but whatever
 nodeindexes == GoToNode))) CYCLE
```

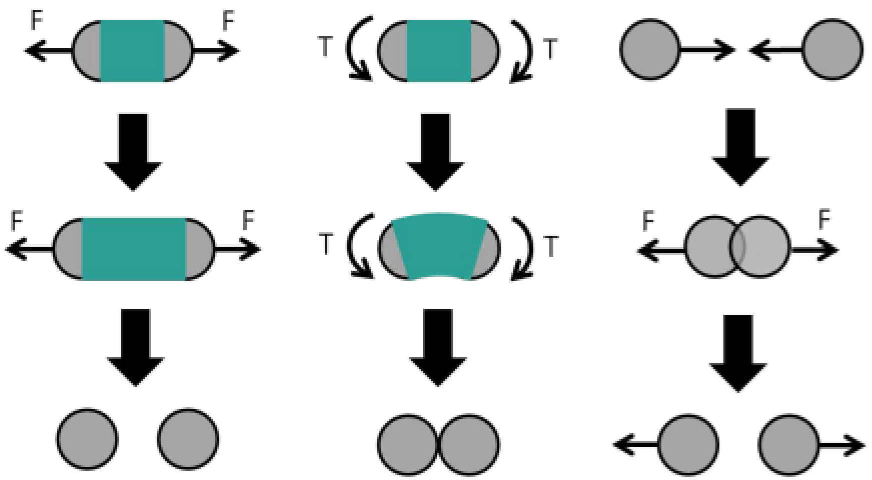
- **Get in touch** – StAndrewsGlaciology.org

St Andrews Glaciology

Coupling between Elmer/Ice and external model

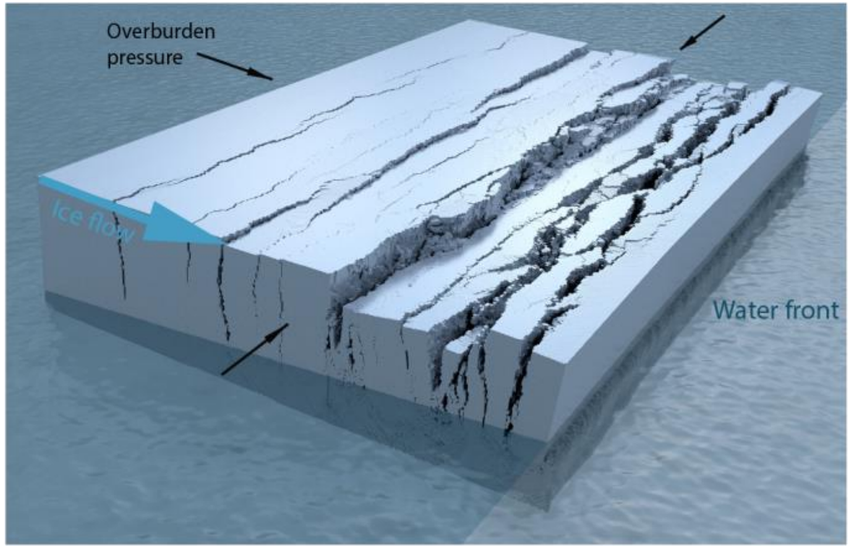
Discrete element model

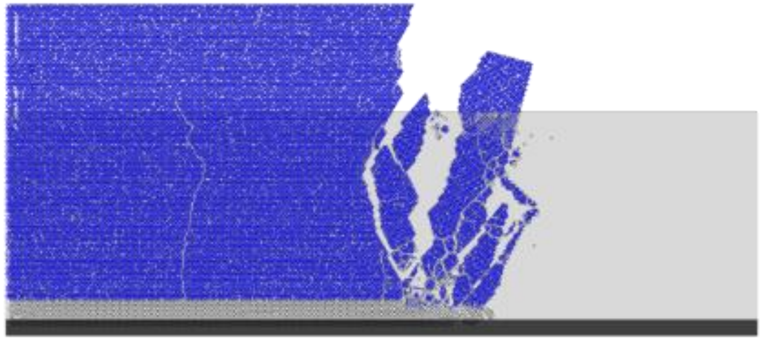
- Numerical particle-based model (Åström et al., 2013) in 2D or 3D
- Glacier divided into discrete particles
- Frozen contacts
  - Beams
  - Inelastic interactions (dissipation of energy)
  - Breaking when elastic load > fracture threshold (stability tune)
- Sliding at the base



Discrete element model

- Is able to use first-principle approach on brittle failure of ice
- Can even include a viscous reaction
- Spatial-scales: resolves glacier in blocks of about Δx =10m length
- Timescales: $\frac{\Delta x}{c} \sim \frac{10}{5000} \sim 10\text{ms}$
- Severe constraints in applicability





Coupled discrete element – continuum model

- Used in a view instances already via offline-coupling, to either evaluate calving behaviour:

  Åström, J.A., D. Vallot, M. Schäfer, E.Z. Welty, S. O'Neel, T.C. Bartholomaus,Yan Liu, T.I. Riikilä, T. Zwinger, J. Timonen, and J.C. Moore, 2014. ***Termini of calving glaciers as self-organized critical systems***, Nature Geoscience, **7**, 874-878, doi:10.1038/ngeo2290

  Benn, D.I., J. Åström, T. Zwinger, J. Todd, F.M. Nick, S. Cook, N.R.J. Hulton, and A. Luckman, 2017. ***Melt-under-cutting and buoyancy-driven calving from tidewater glaciers: new insights from discrete element and continuum model simulations***, Journal of Glaciology, 1-12, doi:10.1017/jog.2017.41.

- Or to determine crevasse positions:

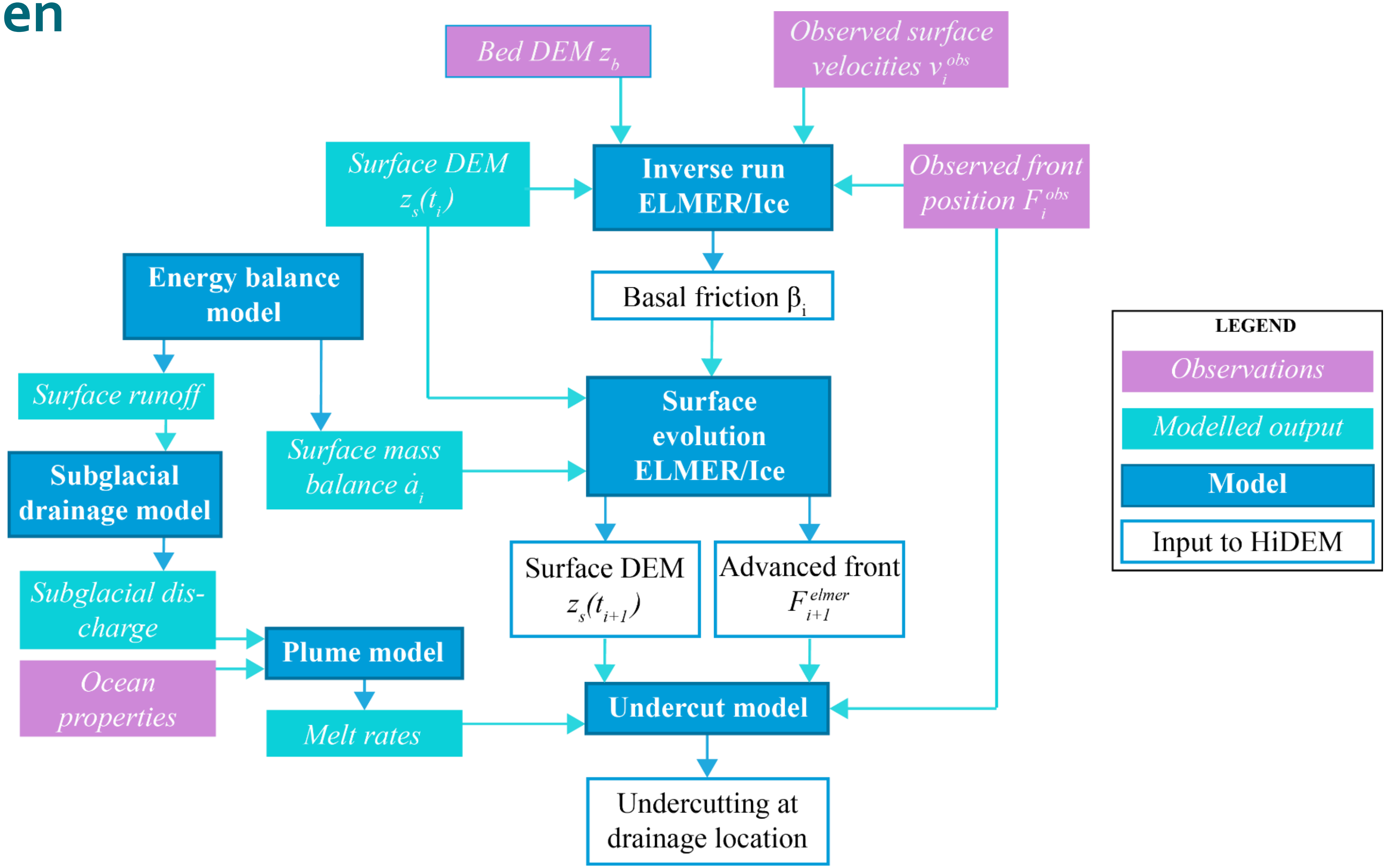
  Gong, Y., T. Zwinger, S. Cornford, R. Gladstone, M. Schäfer, and J.C. Moore, 2016. ***Importance of basal boundary conditions in transient simulations: case study of a surging marine-terminating glacier on Austfonna, Svalbard***, Journal of Glaciology, pp. 1–12, doi:10.1017/jog.2016.121.

Test case: Kronebreen

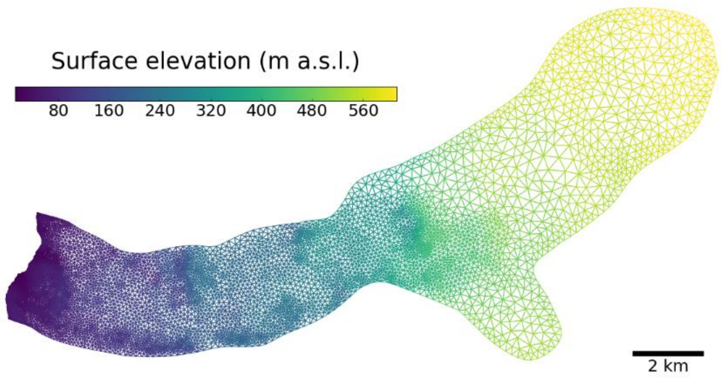
- Tidewater glacier, one of the fastest in Svalbard archipelago
- Sliding at the base
- Started retreating in 2011
- Surface velocity and front positions available for 2014-2015
- High resolution surface and bed topography

Test case: Kronebreen

- Work started by Dorothée Vallot
- Created workflow between Elmer/Ice, surface runoff, basal hydrology, plume model and undercutting
- Currently 2 submitted paper (see next slides)

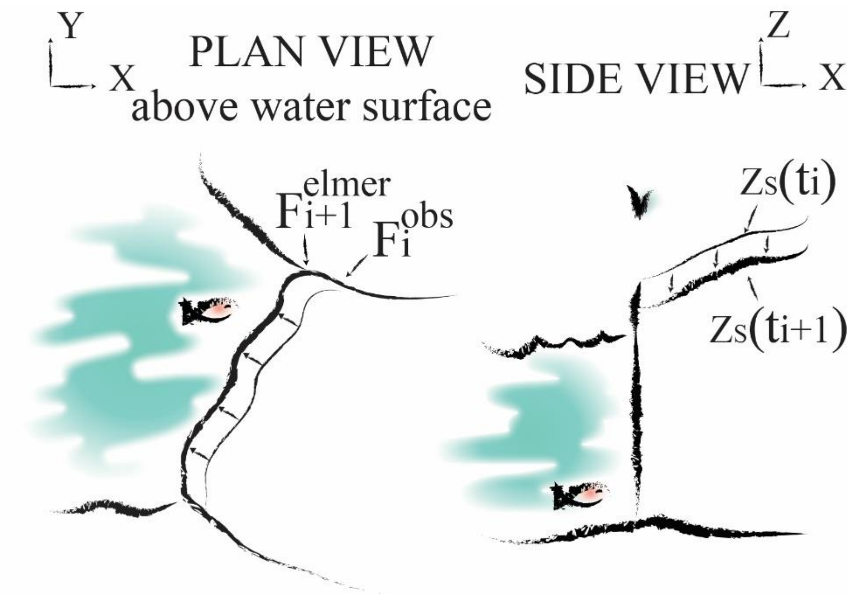


Step 1: Generate the mesh



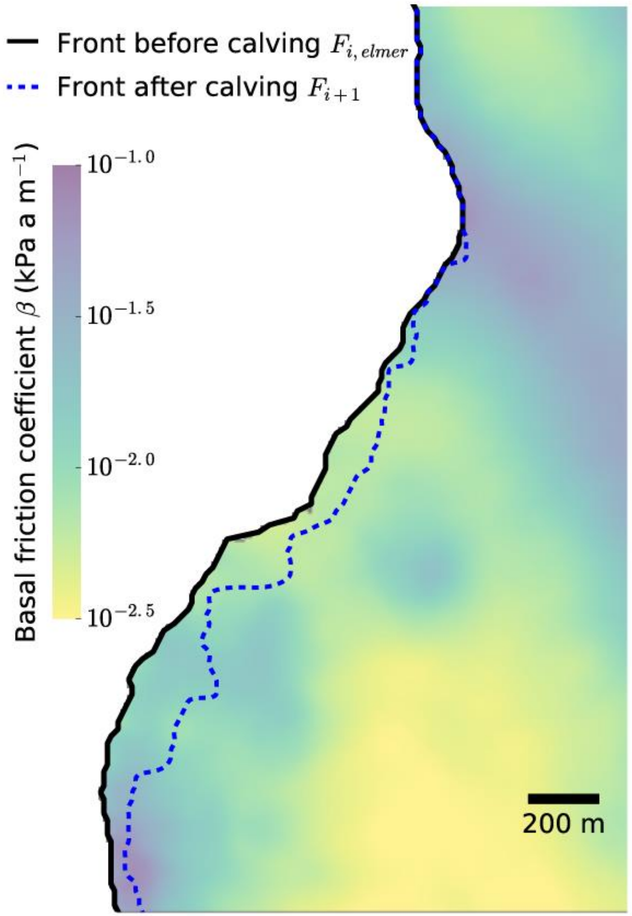
- From front position (initial or modelled) and contour
- Gmsh to create the mesh
- Conversion to Elmer format

Step 2-3: Transient advance with Elmer/Ice and conversion



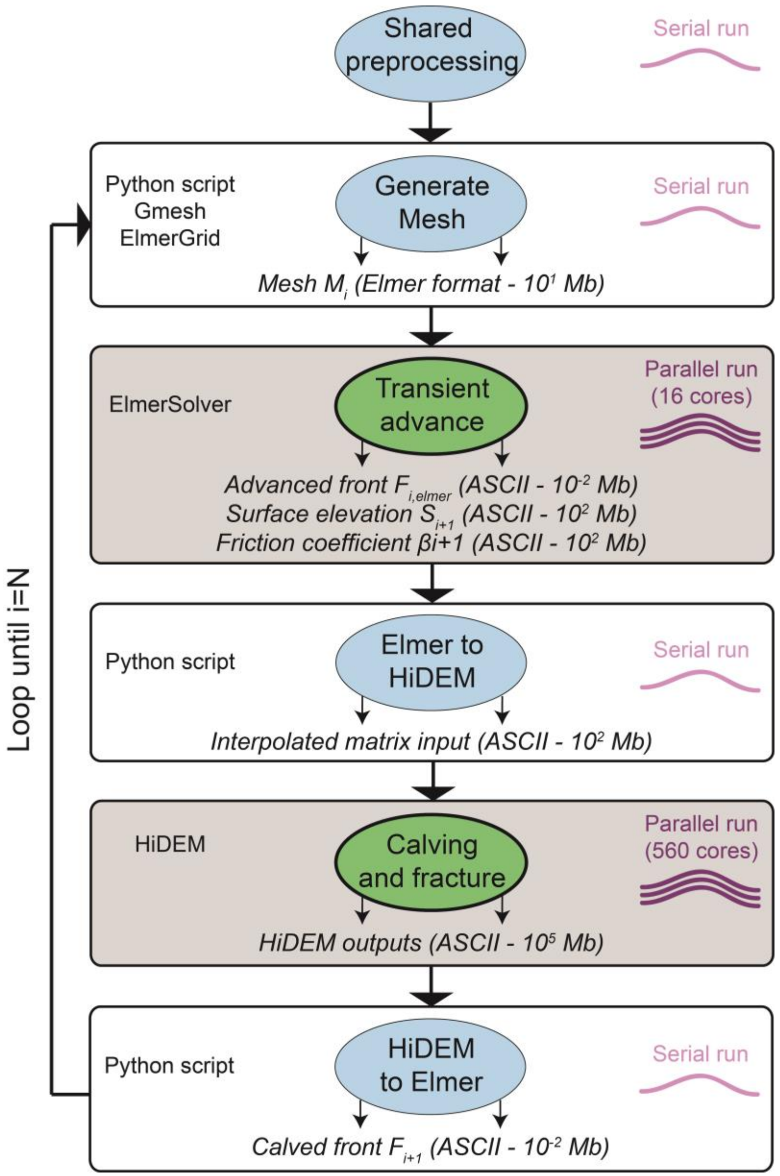
- Stokes equation
- Sliding law
- Surface and front evolution
- Long time period
- Conversion From Elmer/Ice to HiDEM domain

Step 4-5: Calving with HiDEM and new front position



- Scaling o f sliding to accomodate small time step (10^{-4} s)
- New front position to apply to the next step (meshing)

Summary



Shared preprocessing
Serial run
Python script
Gmesh
ElmerGrid
Generate Mesh
Serial run
Mesh M_i (Elmer format - 10^1 Mb)
ElmerSolver
Transient advance
Parallel run (16 cores)
Advanced front $F_{i,elmer}$ (ASCII - 10^{-2} Mb)
Surface elevation S_{i+1} (ASCII - 10^2 Mb)
Friction coefficient βi+1 (ASCII - 10^2 Mb)
Python script
Elmer to HiDEM
Serial run
Interpolated matrix input (ASCII - 10^2 Mb)
HiDEM
Calving and fracture
Parallel run (560 cores)
HiDEM outputs (ASCII - 10^5 Mb)
Python script
HiDEM to Elmer
Serial run
Calved front F_{i+1} (ASCII - 10^{-2} Mb)
Loop until i=N