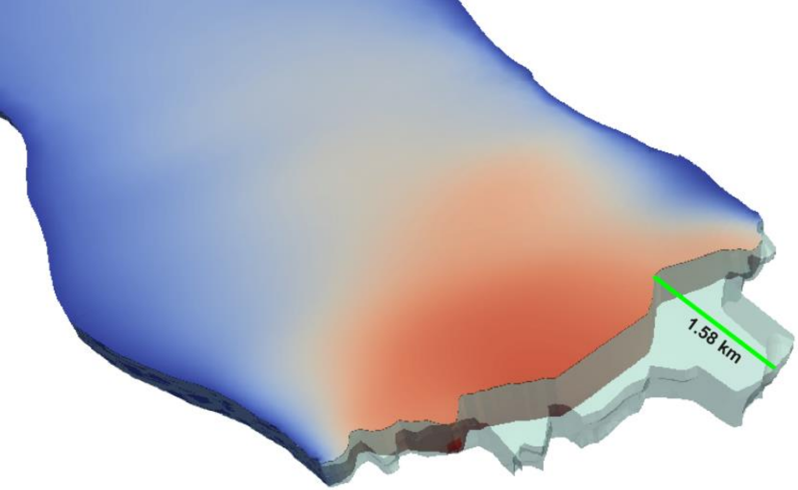


Calving in Elmer/Ice

Joe Todd

St Andrews Glaciology

Scott Polar Research Institute
University of Cambridge

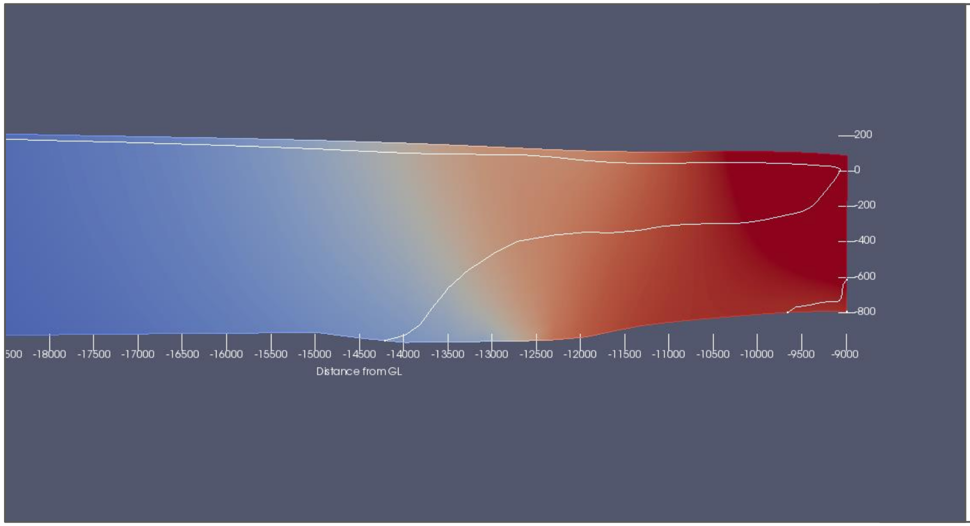
Elmer/Ice Calving Models

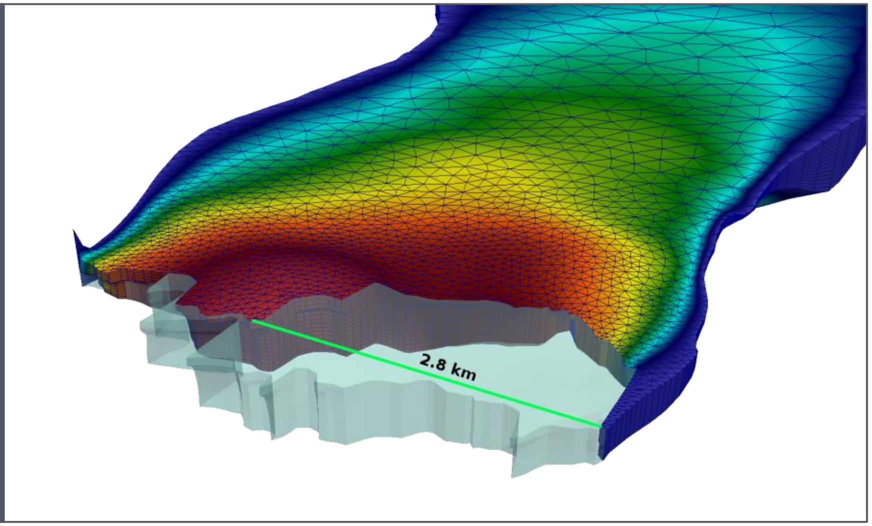
2D Calving:

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

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

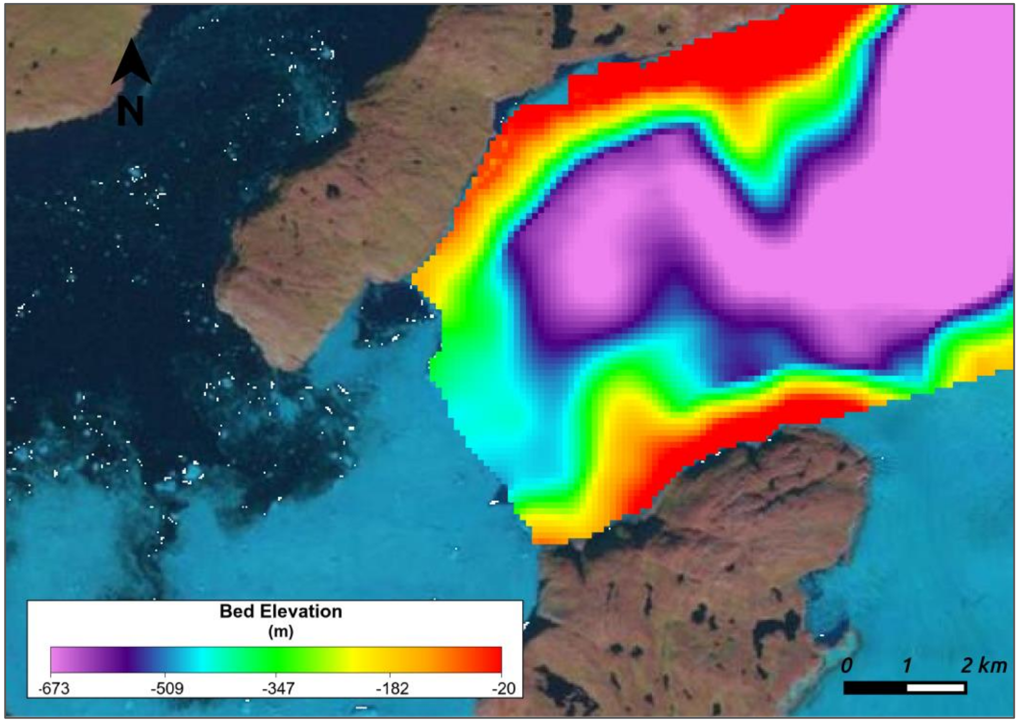
Dependencies

Software:

- GMSH for remeshing
- NETCDF for GridDataReader
- Linux?

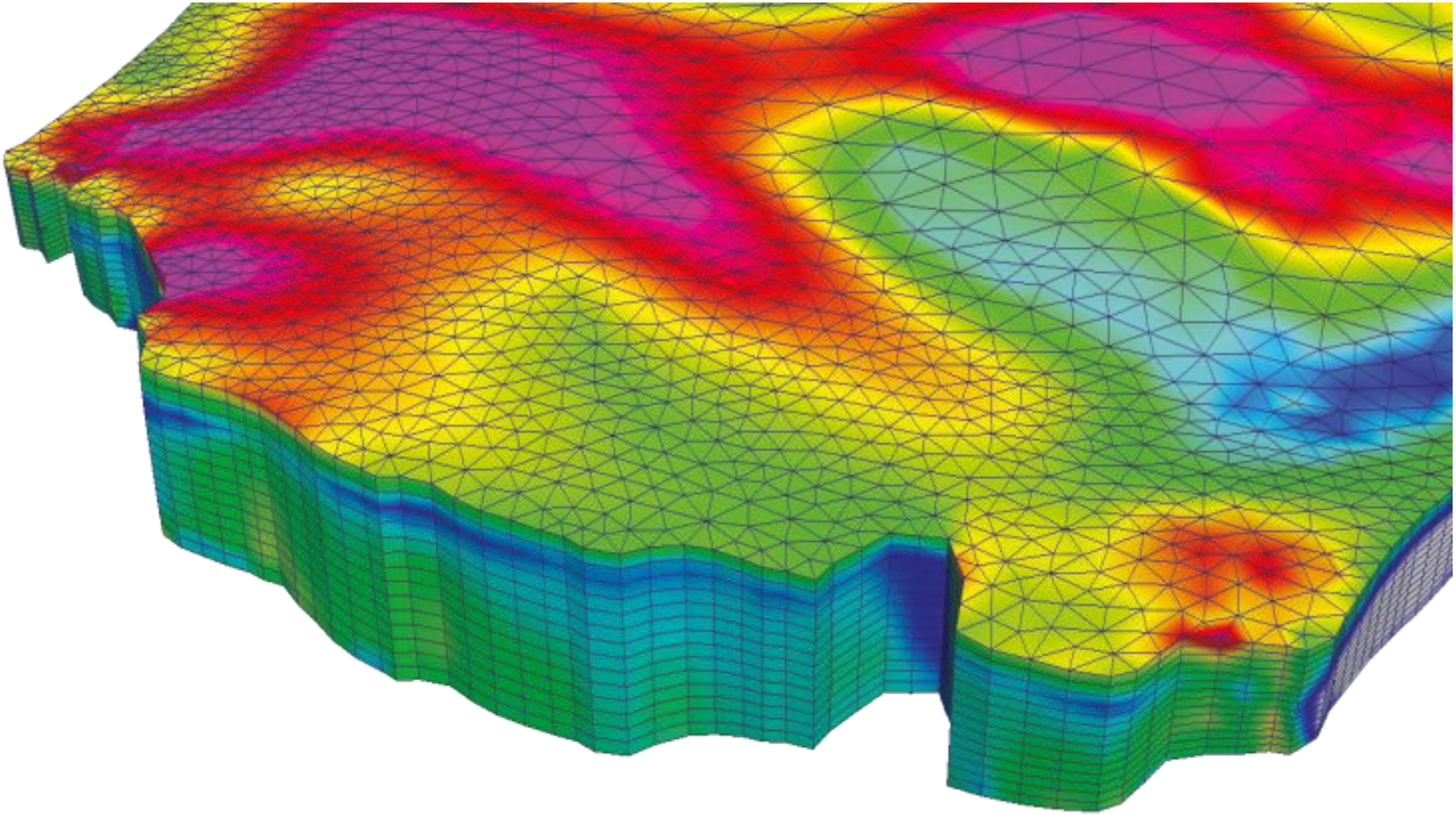
Data:

- Accurate bed topography
- Initial terminus position
- Velocity for inversions

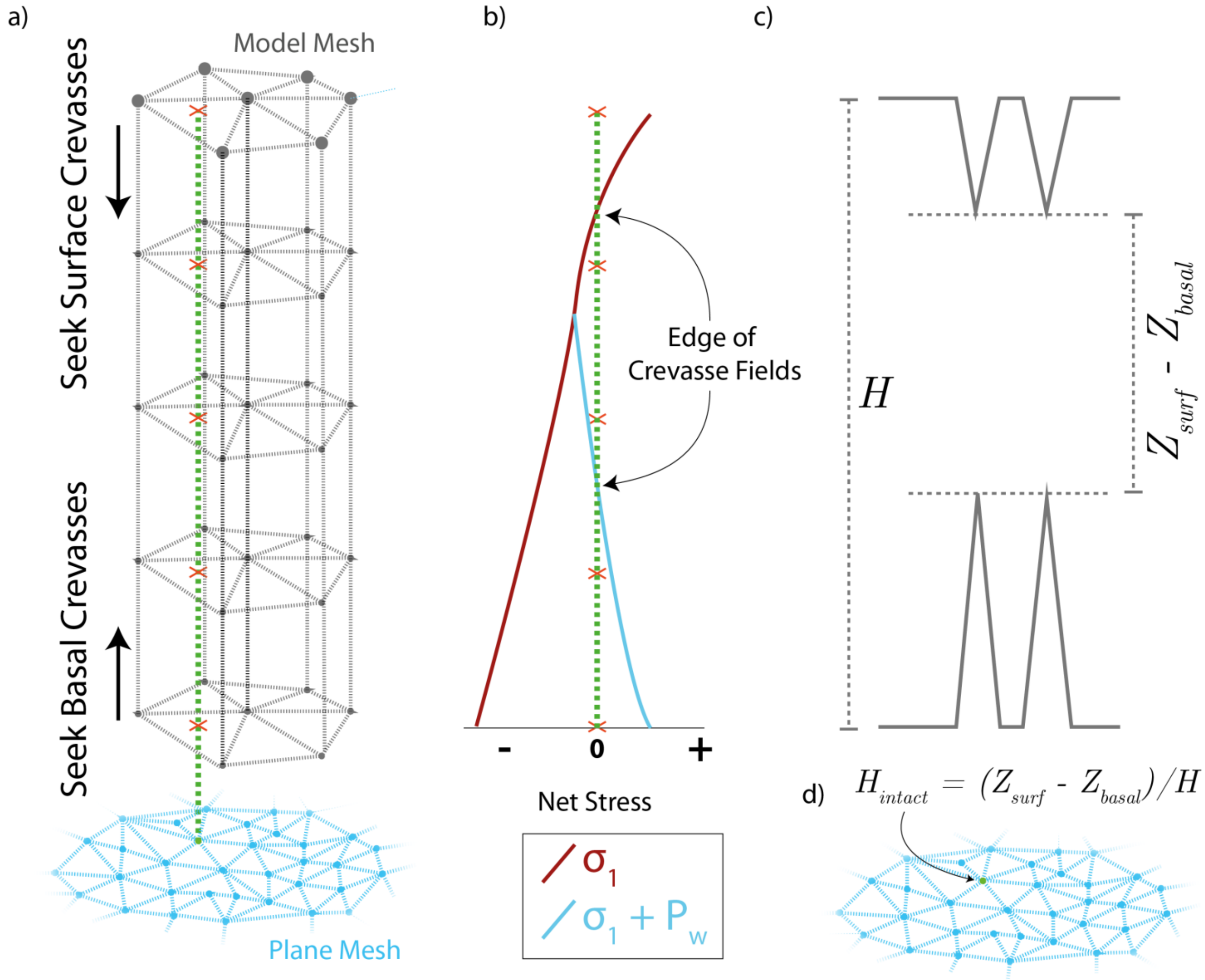


Basal topography produced via mass conservation.

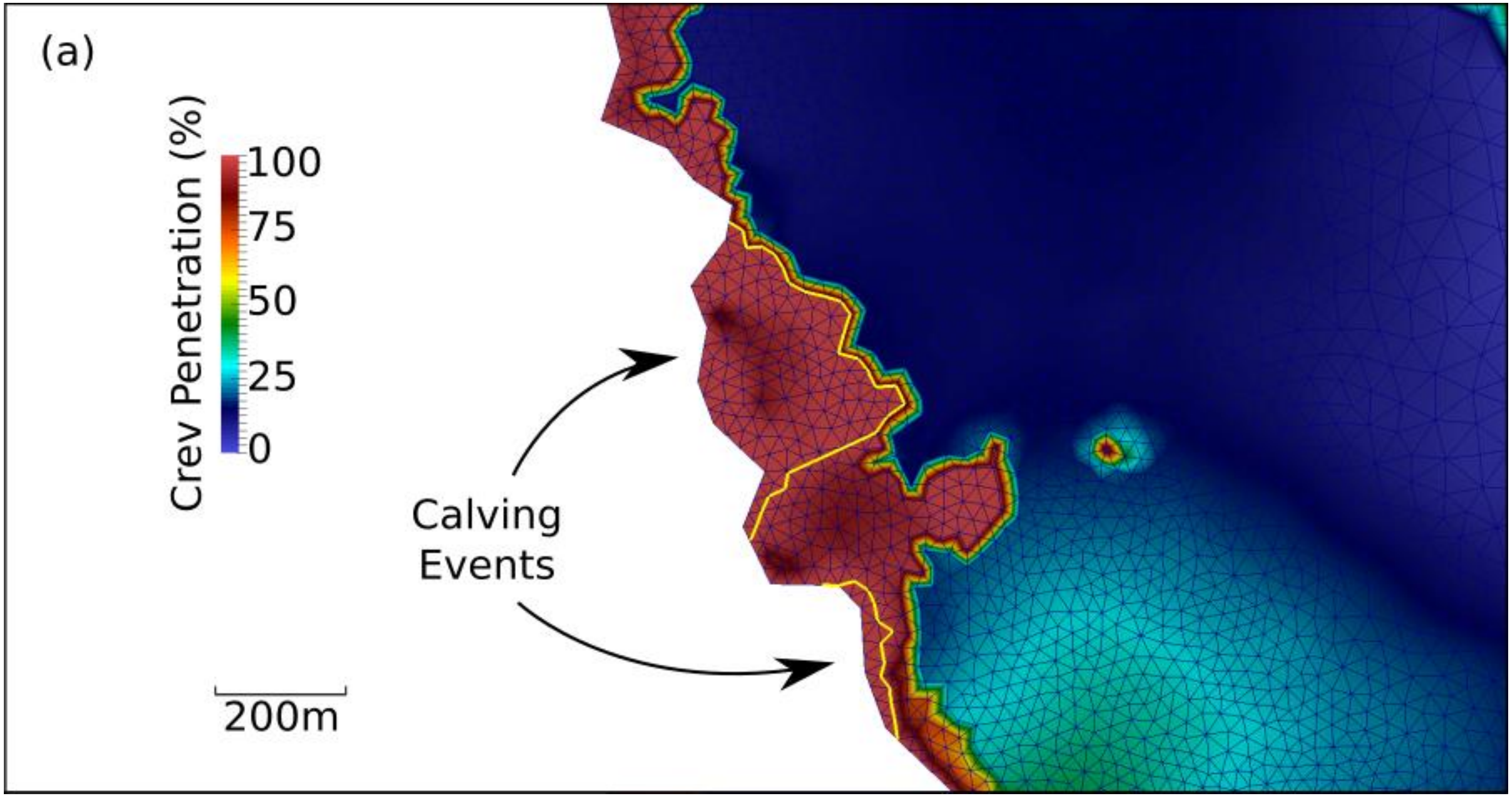
Predicting Calving



Predicting Calving



Predicting Calving



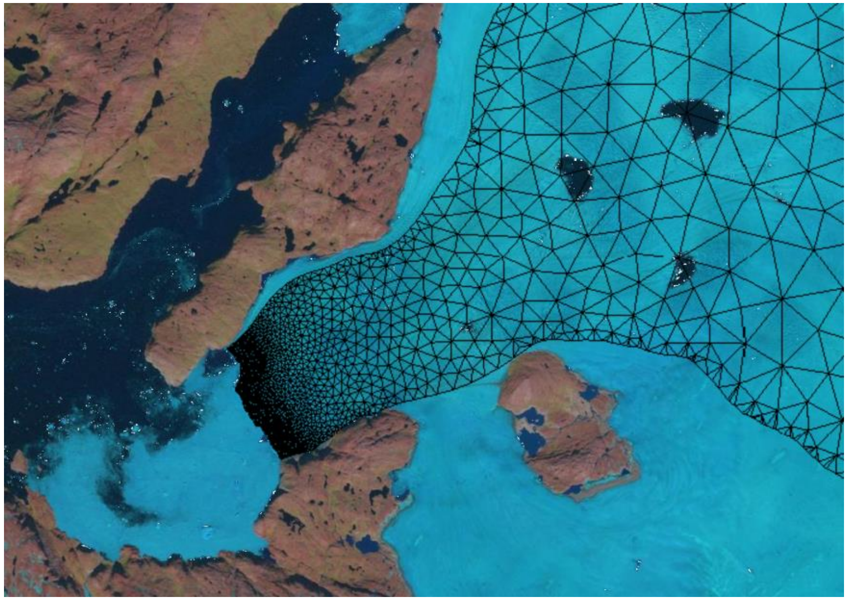
Remeshing

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



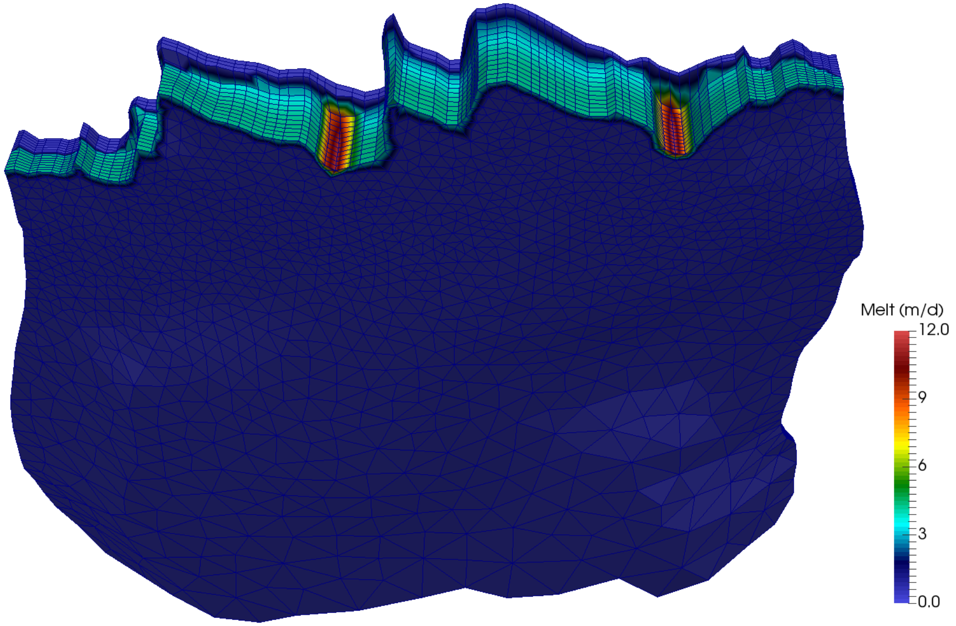
Terminus Advance

Continuous process, unlike calving

FreeSurfaceSolver doesn’t work

CalvingFrontAdvance.F90 computes:

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

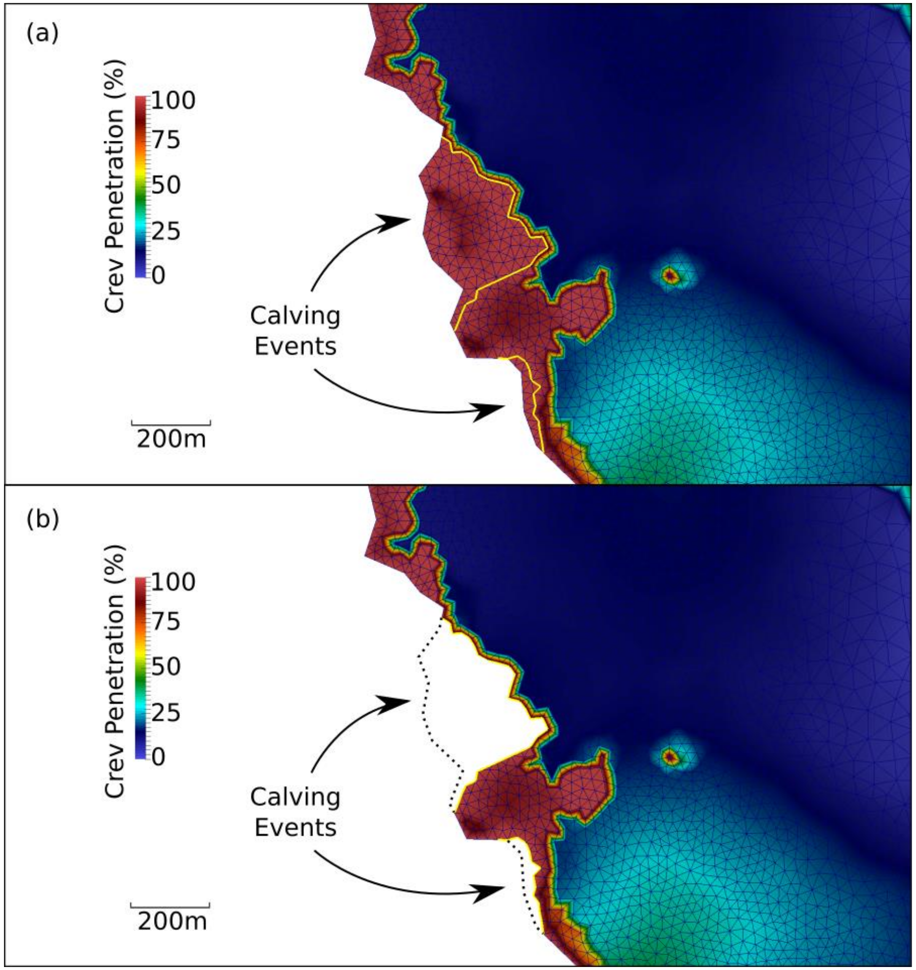


So nodes are free to move in any direction.

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

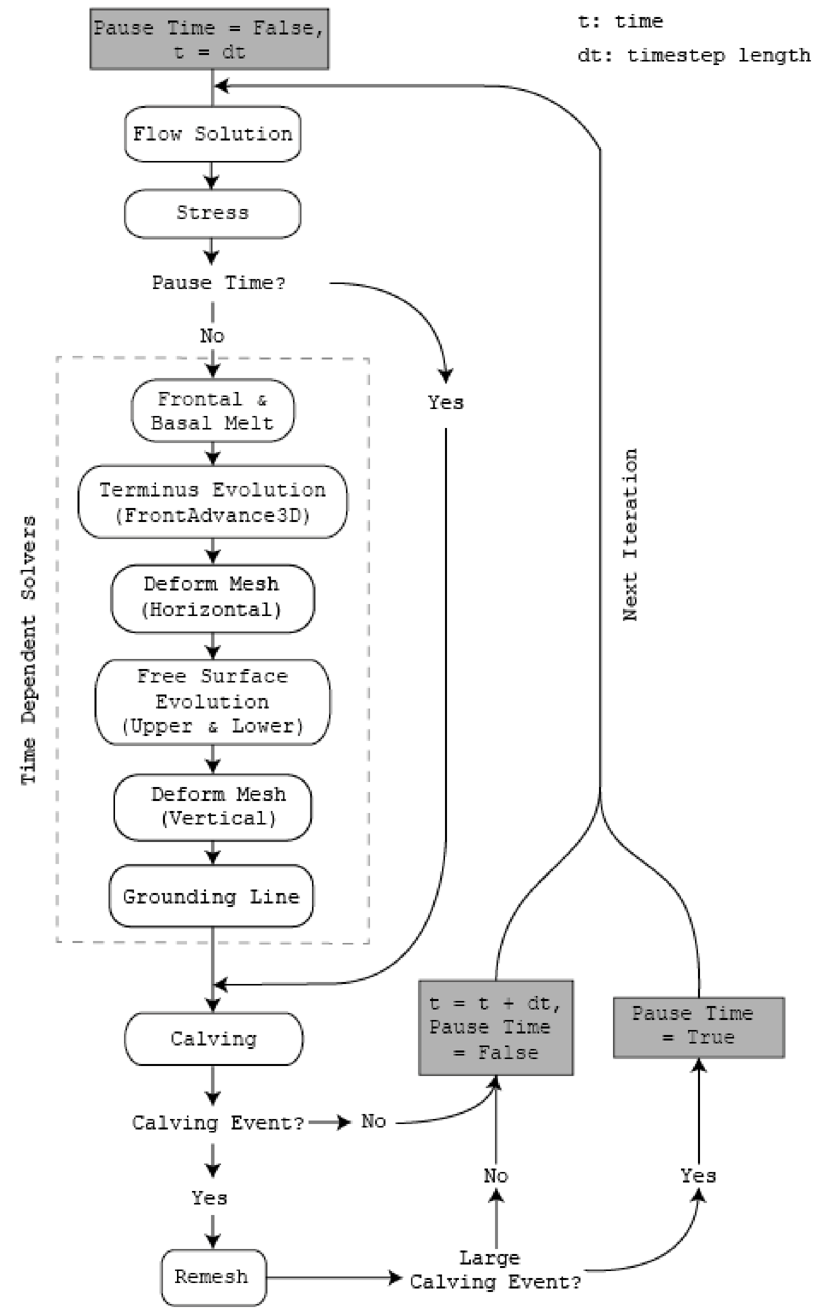
```
Solver 5
  Equation = String "Check NS"
  Procedure = File "ElmerIceSolvers" "CheckFlowConvergence"

  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

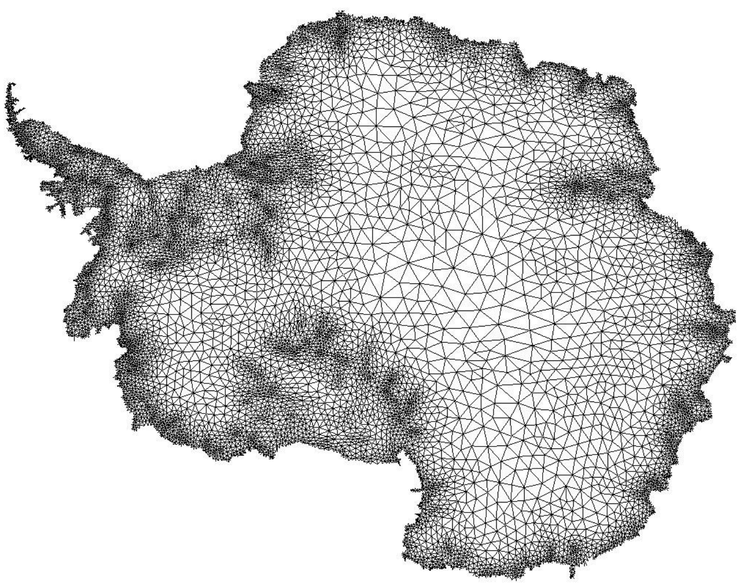
Read the **source code**!

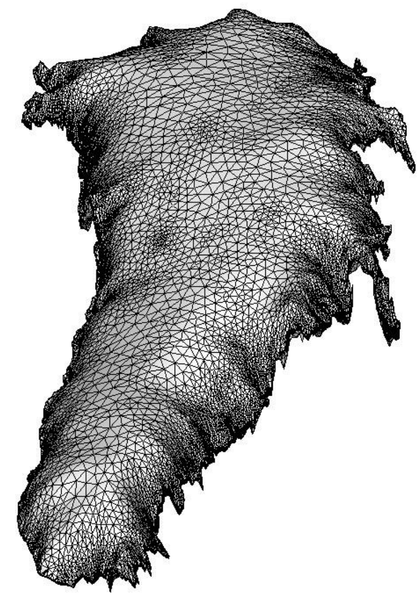
```
tElement % nodeindexes) .GT. 0))) CYCLE
BC elements, stupid way of doing it, but whatever
 nodeindexes == GoToNode))) CYCLE
```

Get in touch – StAndrewsGlaciology.org

Questions about calving in Elmer/Ice?

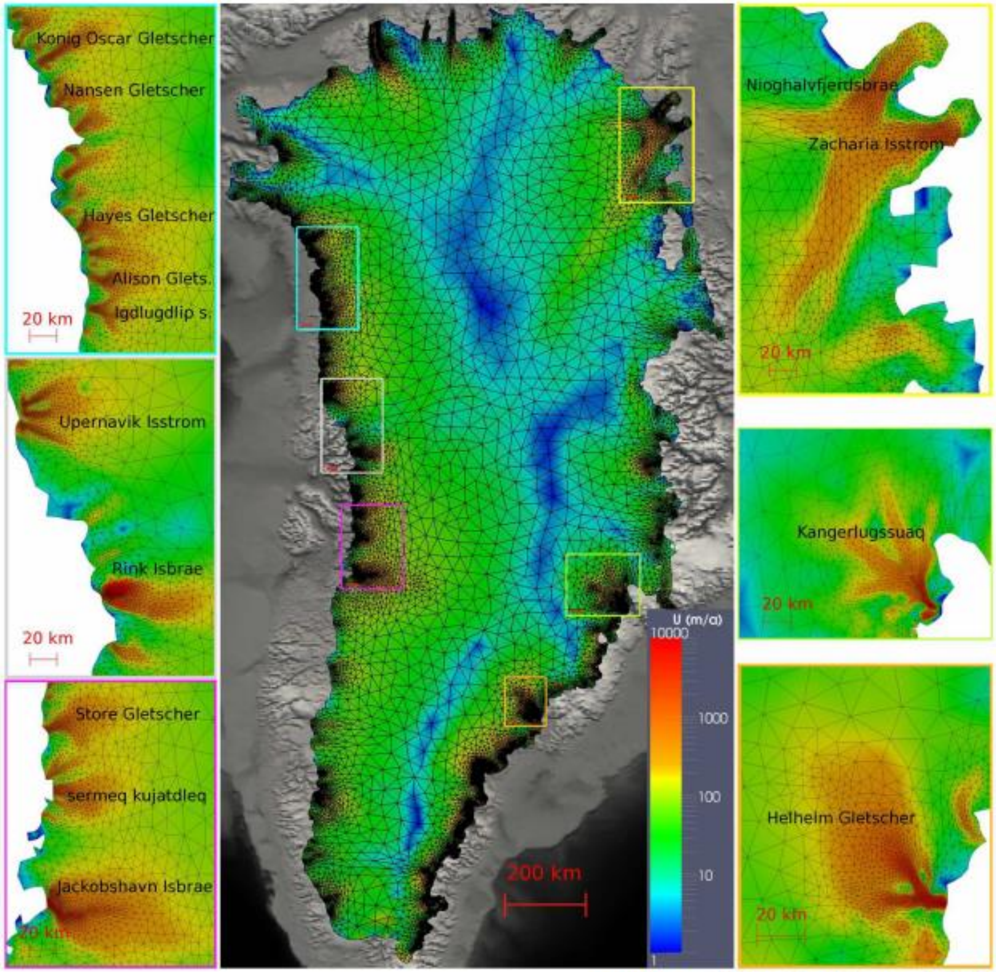
Meshing, Mesh Adaptation & Remeshing





Meshing, Mesh Adaptation & Remeshing

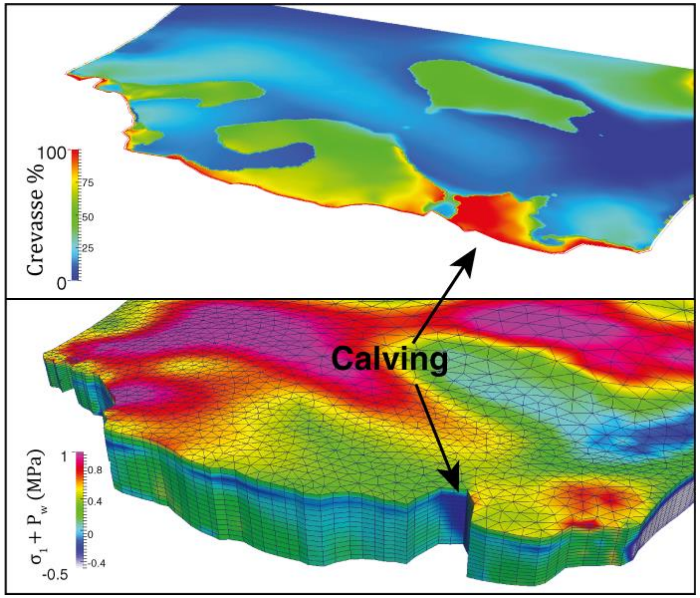
- Remeshing use cases
- Existing remeshing capabilities
- Desired functionality
- Tools



source : Gillet-Chaulet et al., 2012

When do we need to modify meshes?

- When errors are spatially variable (heterogeneous flow, changing boundary conditions)
- When geometry changes (calving, surface adjustment)
- But mesh adaptation is **not** a uniquely glaciological problem!



Glaciology Use Cases

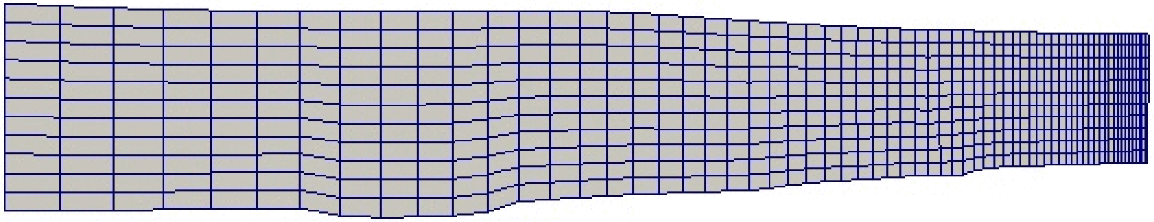
- Surface Adjustment
- Grounding Line Dynamics
- Iceberg Calving

Glaciology typically requires:

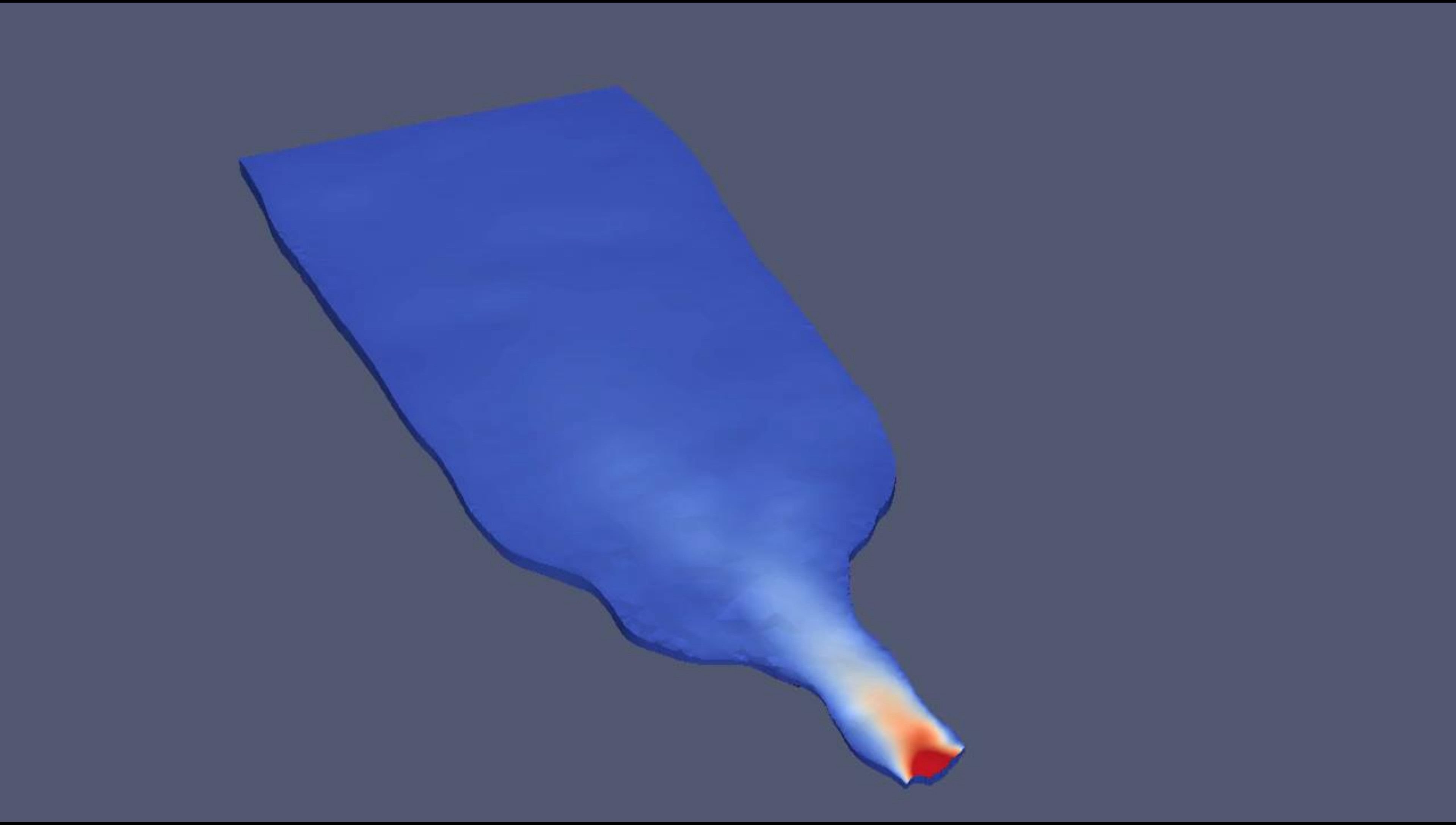
- Mesh Anisotropy
- 3D
- Variable resolution

MeshSolve.F90

- Gradual and simple changes in geometry
- Stretches/squashes mesh
- Mesh topology remains the same (this can be useful)



Complete Remeshing



Complete Remeshing

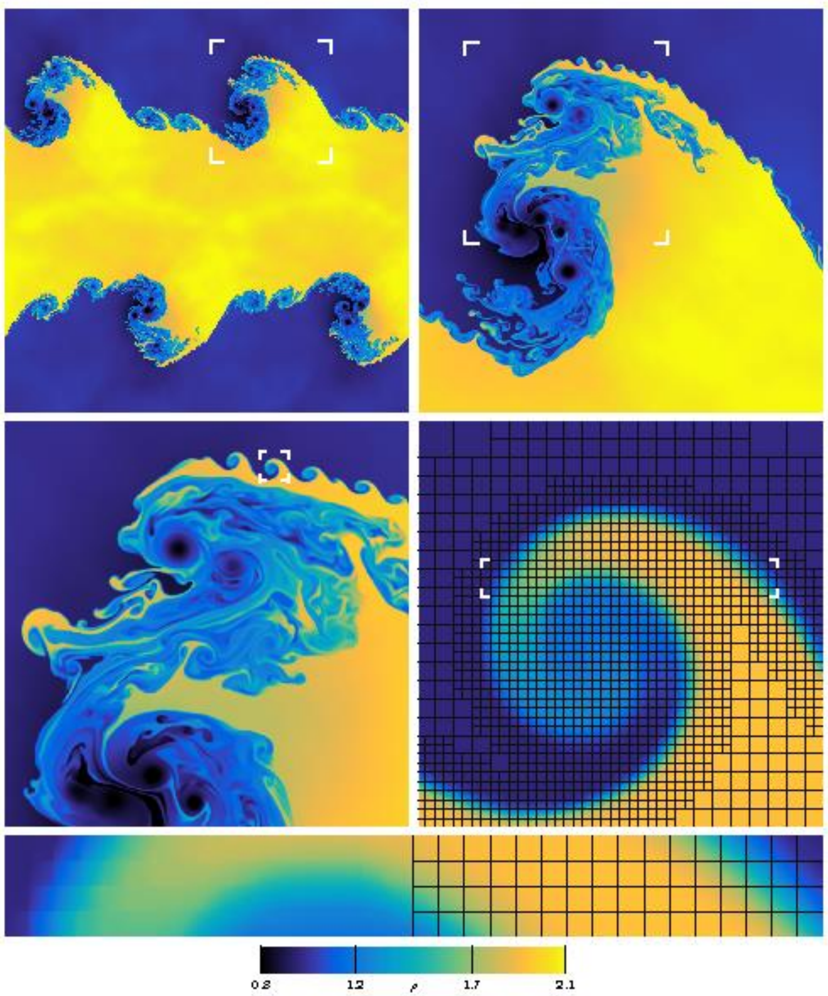
- Ensure high quality mesh even with complex geometry changes
- Control over mesh resolution through entire simulation

But…

- Relies on external tools
    - Command line call to GMSH - bit messy
- Makes data analysis a bit trickier
- Only able to produce structured meshes

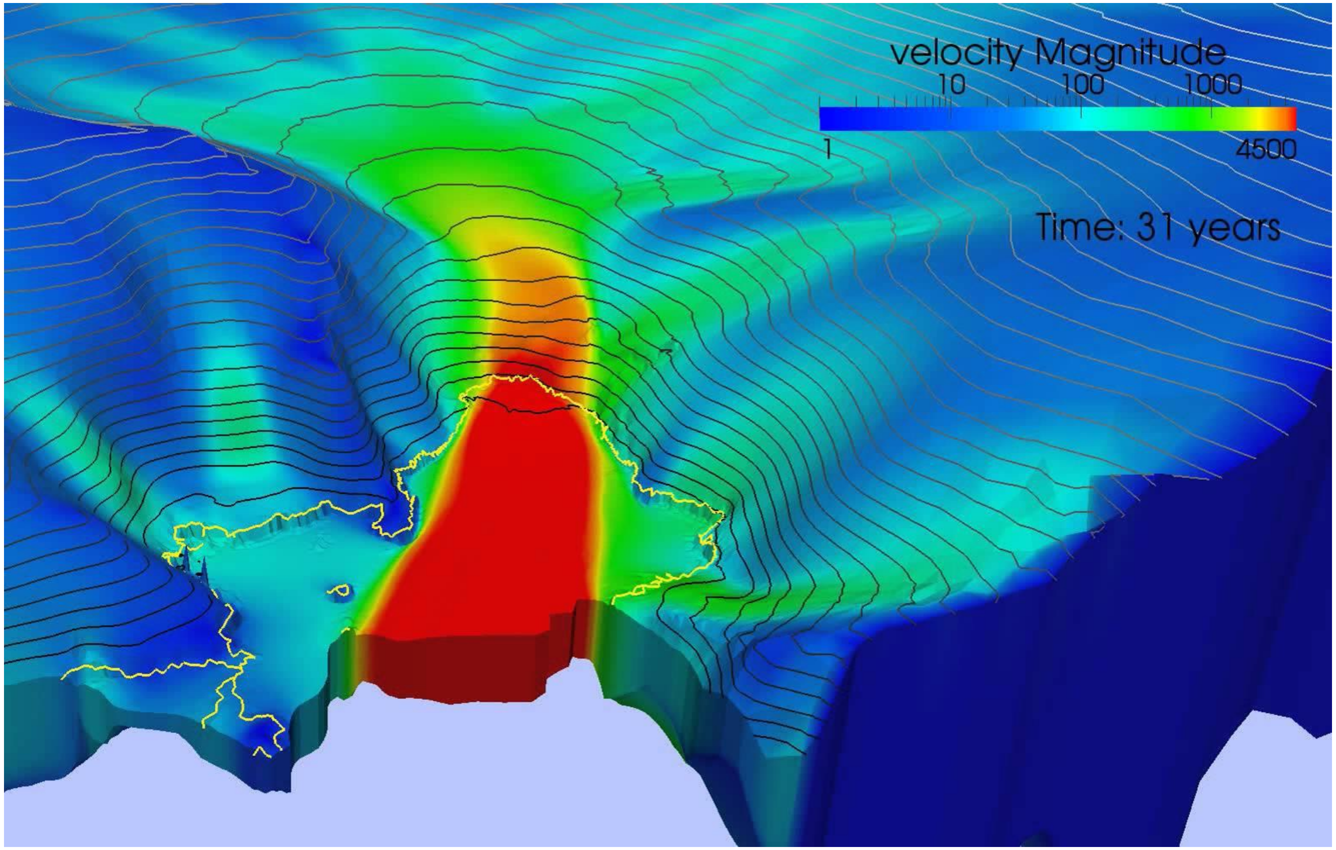
Mesh Refinement

- Error reduction scheme
- Split elements based on metric
  - Computed error
  - Solution gradient
  - Distance from grounding line
- No geometry changes
- Can also join elements if error is low for computational efficiency



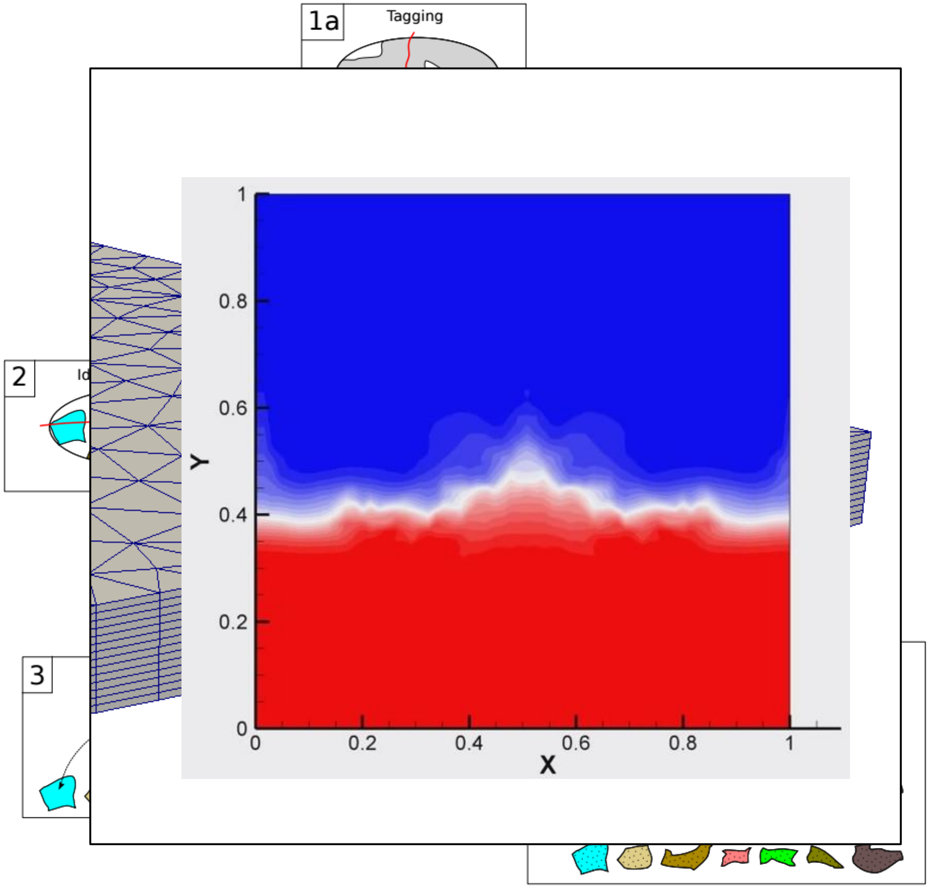
Modelling Ice Sheets

- Evolving surfaces
- Evolving terminus/ice shelf extents
- Evolving grounding lines
- Evolving basal slip?



Issues

- Performance bottlenecks
  - Go parallel?
- Need for anisotropy
- Structured/Unstructured
  - What’s the advantage of structured?
- Numerical Diffusion
  - Can this be avoided/minimised?



Available Tools

GMSH – weapon of choice for most Elmer users?

YAMS – surface meshes only

Mmg/PaMPA

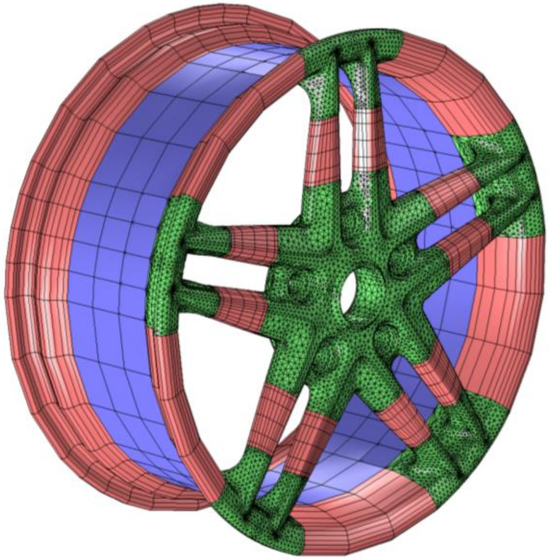
- parallel 3D remeshing
- libs with FORTRAN interface!

Elmer internal functionality:

- Mesh extrusion
- Structured mesh mapping
- Adaptive splitting

Priorities for Future Work?

- Fabien has developed solvers for mesh refinement using Mmg
    - Are we done with YAMS?
- Joe has developed pseudo-structured remeshing
- Lots of interest in GL refinement
- Selective partial remeshing – minimise numerical diffusion?
- Let’s collaborate and use libraries when possible!
- Avoid re-inventing the wheel



Parallel adaptivity

Some thoughts from Elmer project manager (Peter Råback)

What we currently use:
 - Error indicator exists for a few equations (problem specific indicators)
 - Adaptivity can be done either by splitting or external mesher
 - Splitting is done on edges and results to mesh ratio of two and funny looking interfaces.
 - External mesher needs an indicator for the mesh density. This was previously done for ElmerMesh2D and it could probably be done with some other meshers as well.

Unfortunately the adaptivity is never parallel in Elmer. So to go into parallel adaptivity some other steps should be done in parallel within ElmerSolver: partitioning.

Parallel adaptivity

Partitioning in ElmerSolver:

- There exists a PartitionMesh solver and six test cases InternalPartitioning*
- PartitionMesh solver is designed to be more content aware so it can make good decisions based on the sif
- Currently supports only some geometric routines of my own. Also supports hybrid methods where parts of the mesh are partitioned hierarchically with different methods. - Also Metis/Scotch should be implemented.
- Currently saves mesh to disk in partitioned format (WriteMeshToDiskPartitioned).
- Mesh distribution by a master process to subprocesses should be implemented
- When this would work as a solver the features could be merged into the library more closely.

When we can partition meshes inside ElmerSolver it is much easier to adapt further adaptive stategies. Personally, I'm little bit hesitant about the local partitioning schemes. It could be better to have global remeshing that would be implemented also as a 2D master process on one partition.