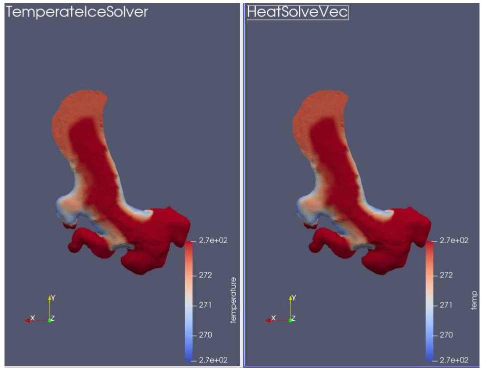
Code/Method updates

Peter Råback and Thomas Zwinger
via Zoom

2.2.2022

Vectorized Heat Transfer Solver

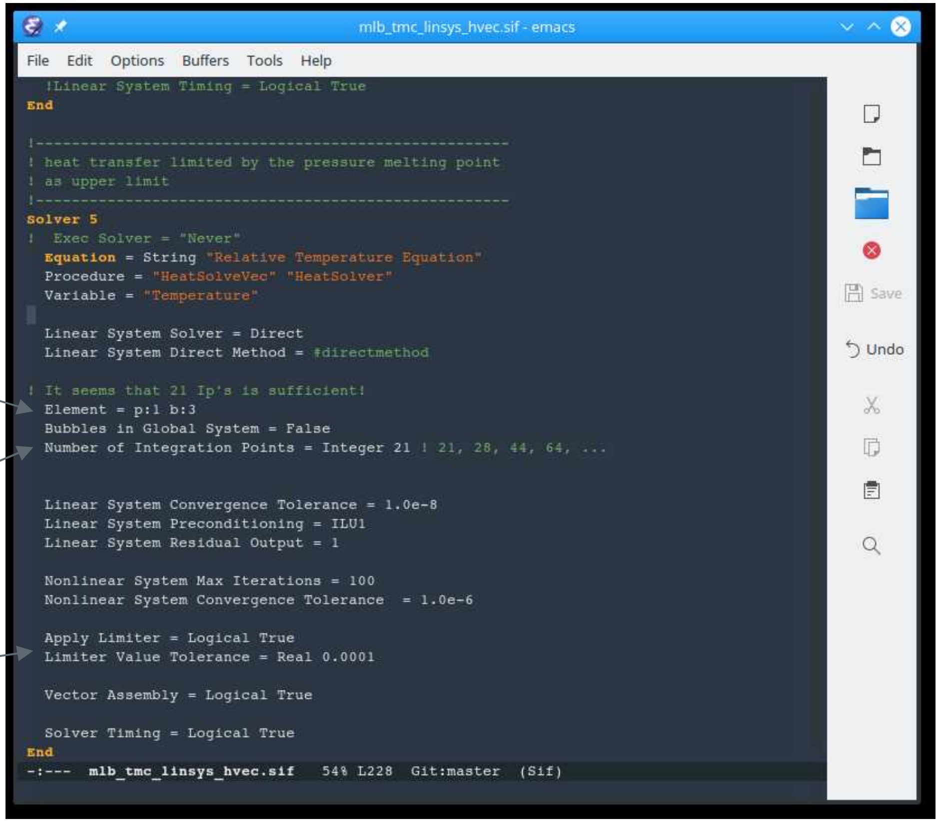


Vectorized Heat Transfer Solver

- We needed to add features to heat equation
  - Old HeatSolve not easily modified for the intended use
  - New modern of HeatSolver written to gradually replace the old
- Features of the new solver
  - Vectorized & threaded
  - Able to deal with discontinuities
- Still missing some features from old solver
  - Phase change, compressibility, heat control

Vectorized Heat Transfer Solver

- `HeatSolveVec` uses OpenMP SIMD + threading for assembly
- Bubble stabilization
  - Automatic selection of bubble degrees if not set or mixed element mesh
- Use larger amount if IP points to fill vector units
- Use library functionality for pressure melting point limit



Vectorized Heat Transfer Solver

- Upper limit has to be given in **Body Force** rather than Material
- Use normal function for material parameters
- Same for boundary conditions

```
Body Force 1
 Temperature Upper Limit = Real 273.15 ! we
ignore pressure melting point

Material 1
  Heat Capacity = Variable Temperature
  Real lua "capacity(tx[0])*yearinsec^(2.0)"
  Heat Conductivity = Variable Temperature
  Real lua "conductivity(tx[0])*yearinsec*Pa2MPa"

Boundary Condition 3
  Name = "bedrock"
  Heat Flux = Real #0.050 * yearinsec * Pa2MPa
```

Vectorized Heat Transfer Solver

Benchmarks **`mlb_tmc_linsys(_hvec).sif`** on 6 core Intel i5-9400F

- **Whole run on 25 m resolution mesh:**
- HeatSolveVec:

```
SOLVER TOTAL TIME(CPU,REAL):        231.89
234.15
```

- TemperateIceSolver:

```
SOLVER TOTAL TIME(CPU,REAL):        345.78
351.73
```

- ~2/3rd of runtime

- **Whole run on 50 m resolution mesh**:
- HeatSolveVec :

```
SOLVER TOTAL TIME(CPU,REAL):        43.24
44.07
```

TemperateIceSolver:

```
SOLVER TOTAL TIME(CPU,REAL):        64.27
65.86
```

- ~2/3rd of runtime

Vectorized Heat Transfer Solver

Benchmarks **`mlb_tmc_linsys(_hvec).sif`** 50m on 6 core Intel i5-9400F

- **Solver timing run on 50 m resolution mesh:**

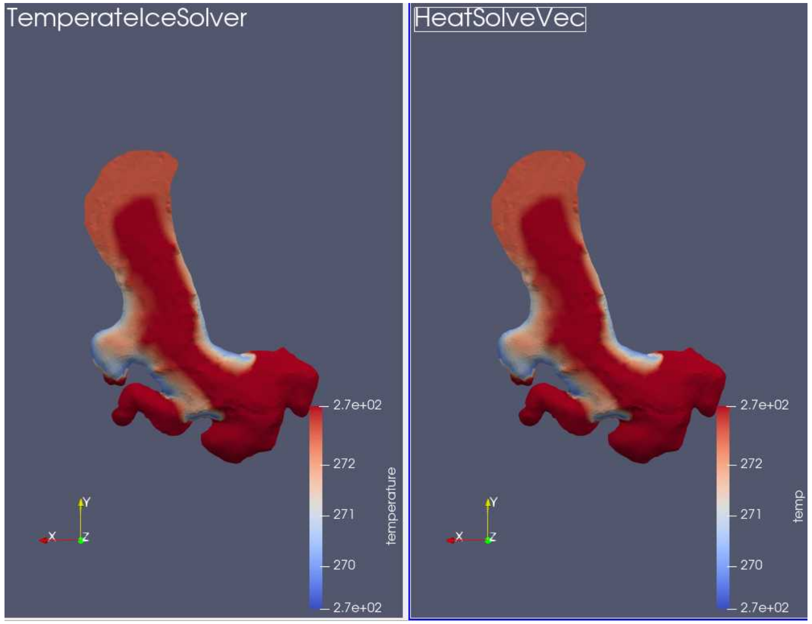
`TemperateIceSolver:`

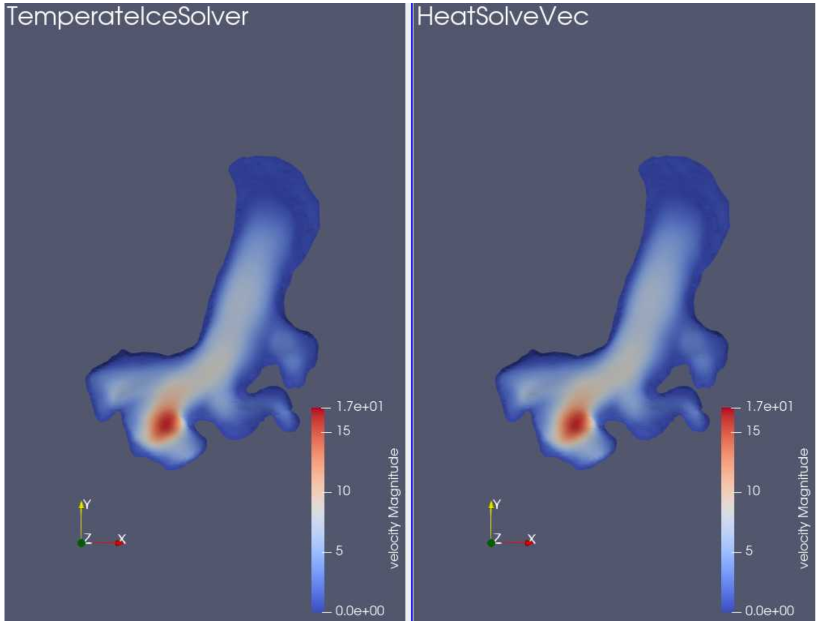
```
(CPU,REAL):   4.80   4.97 (s)
(CPU,REAL):   4.75   4.94 (s)
(CPU,REAL):   4.70   4.91 (s)
(CPU,REAL):   4.72   4.90 (s)
(CPU,REAL):   4.81   4.95 (s)
(CPU,REAL):   4.69   4.90 (s)
```

`HeatSolveVec:`

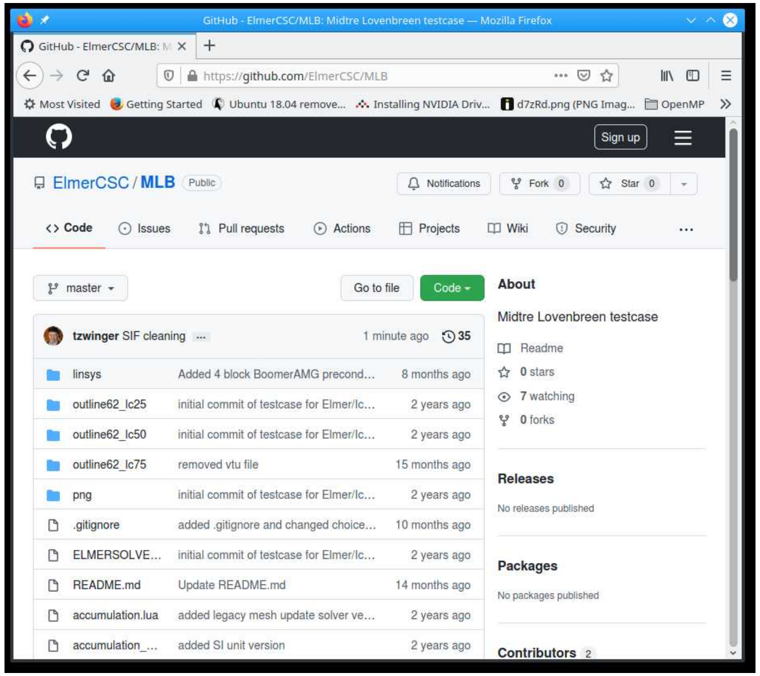
```
(CPU,REAL):   2.61   2.75 (s)
(CPU,REAL):   1.59   1.64 (s)
(CPU,REAL):   1.59   1.63 (s)
(CPU,REAL):   1.05   1.10 (s)
(CPU,REAL):   0.53   0.54 (s)
(CPU,REAL):   0.54   0.55 (s)
```

Vectorized Heat Transfer Solver



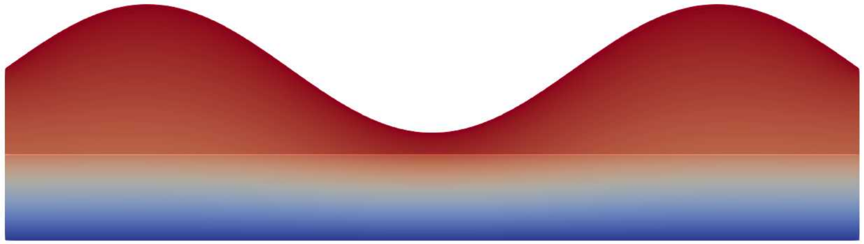


Vectorized Heat Transfer Solver



- https://github.com/ElmerCSC/MLB
- Testcase:
  `mlb_tmc_linsys_hvec.sif`

Vectorized Heat Transfer Solver

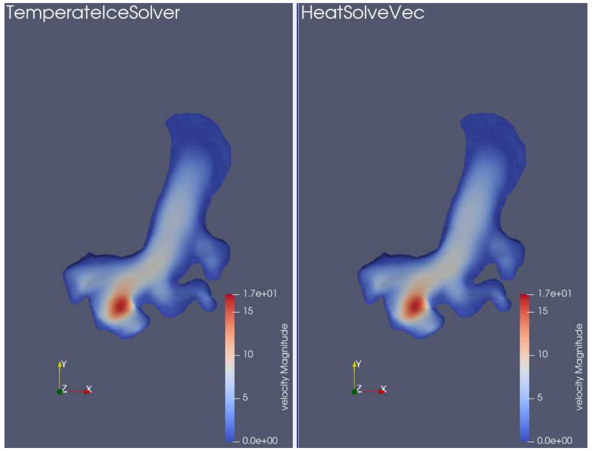


Test case: DiscontinuousTempSlabDG

```
Solver
! Here we define the basis
  Discontinuous Galerkin = Logical True
  DG Reduced Basis = Logical True
  DG Reduced Basis Master Bodies(1) = 1
```

```
Boundary Condition
! Jump condition
  Heat Gap = Logical True
  Heat Gap Coefficient = Real 1.0e1
```

Bug-fix: Block pre-conditioner



Block-preconditioner in `IncompressibleNSVec`

- Stokes problem block-structure

$$\begin{bmatrix} \mathbf{A} & \mathbf{B}^T \\ \mathbf{B} & \mathbf{C} \end{bmatrix} \begin{bmatrix} \mathbf{V} \\ \mathbf{P} \end{bmatrix} = \begin{bmatrix} \mathbf{F} \\ \mathbf{G} \end{bmatrix}$$

from stabilization

- Optimal pre-conditioner with Pressure-Schur complement, $\mathbf{Q}$,

$$\mathbf{P} = \begin{bmatrix} \mathbf{A} & \mathbf{B}^T \\ \mathbf{0} & \mathbf{Q} \end{bmatrix}$$

  - Either split velocity block, **A**, into 3x3 (recommended!)

  `Block Structure(4)=Integer 1 2 3 4`

  - Or as one

  `Block Structure(4)=Integer 1 1 1 4`

```
Linear System Solver = "Block"
Block Gauss-Seidel = Logical True
Block Matrix Reuse = Logical False
Block Scaling = Logical False
Block Preconditioner = Logical True
! Default is [1 2 3 4]
Block Structure(4) = Integer 1 2 3 4
!  Block Order(2) = Integer 2 1
! Linear System Scaling = False
! Linear system solver for outer loop
!-----------------------------------------
  Outer: Linear System Solver = "Iterative"
  Outer: Linear System Iterative Method = GCR
  Outer: Linear System GCR Restart =  250
  Outer: Linear System Residual Output = 1
  Outer: Linear System Max Iterations = 200
  Outer: Linear System Abort Not Converged = False
  Outer: Linear System Convergence Tolerance = 1e-8
```

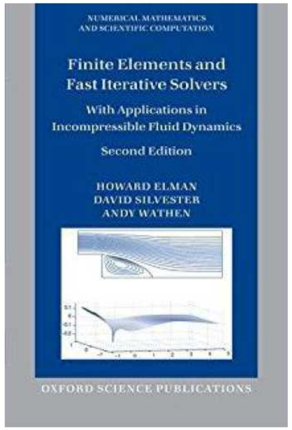
Block-preconditioner in `IncompressibleNSVec`

- Inner solutions (of blocks)
- Blocks 1,2,3 here associated with velocity components 1,2,3

$$\mathbf{P} = \begin{bmatrix} \begin{matrix} \mathbf{A}_1 & \mathbf{0} & \mathbf{0} \\ \mathbf{A}_{12} & \mathbf{A}_2 & \mathbf{0} \\ \mathbf{A}_{31} & \mathbf{A}_{23} & \mathbf{A}_3 \end{matrix} & \mathbf{0} \\ \mathbf{0} & \mathbf{Q} \end{bmatrix}$$

- Block 4 associated with pressure (preconditioned with scaled mass matrix is suggested by Elman)

$$\mathbf{A}_{44} = \mathbf{Q} = \mu^{-1}\mathbf{1}$$




```
block 11: Linear System Convergence Tolerance = $blocktol
block 11: Linear System Solver = "iterative"
block 11: Linear System Scaling = false
block 11: Linear System Preconditioning = ilu
block 11: Linear System Residual Output = 100
block 11: Linear System Max Iterations = 500
block 11: Linear System Iterative Method = idrs

block 22: Linear System Convergence Tolerance = $blocktol
block 22: Linear System Solver = "iterative"
block 22: Linear System Scaling = false
block 22: Linear System Preconditioning = ilu
block 22: Linear System Residual Output = 100
block 22: Linear System Max Iterations = 500
block 22: Linear System Iterative Method = idrs

block 33: Linear System Convergence Tolerance = $blocktol
block 33: Linear System Solver = "iterative"
block 33: Linear System Scaling = false
block 33: Linear System Preconditioning = ilu
block 33: Linear System Residual Output = 100
block 33: Linear System Max Iterations = 500
block 33: Linear System Iterative Method = idrs

block 44: Linear System Convergence Tolerance = $blocktol
block 44: Linear System Solver = "iterative"
block 44: Linear System Scaling = true
block 44: Linear System Preconditioning = ilu
block 44: Linear System Residual Output = 100
block 44: Linear System Max Iterations = 500
block 44: Linear System Iterative Method = idrs
```

Block-preconditioner in `IncompressibleNSVec`

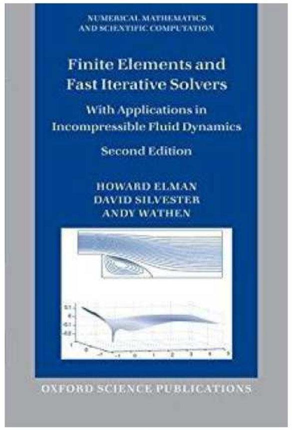
- Inner solutions (of blocks)
- Block 1 here associated with combined velocity components 1,2,3 and solved as a single block

$$\mathbf{P} = \begin{bmatrix} \mathbf{A}_{11} & \mathbf{0} \\ \mathbf{0} & \mathbf{Q} \end{bmatrix}$$

- Block 2 associated with pressure (preconditioned with scaled mass matrix is suggested by Elman)

$$\mathbf{A}_{22} = \mathbf{Q} = \mu^{-1}\mathbf{1}$$

- There was a bug that prohibited non-square sub-blocks to be computed correctly.




```
block 11: Linear System Convergence Tolerance = $blocktol
block 11: Linear System Solver = "iterative"
block 11: Linear System Scaling = false
block 11: Linear System Preconditioning = ilu0
block 11: Linear System Residual Output = 100
block 11: Linear System Max Iterations = 500
block 11: Linear System Iterative Method = idrs


block 22: Linear System Convergence Tolerance = $blocktol
block 22: Linear System Solver = "iterative"
block 22: Linear System Scaling = true
block 22: Linear System Preconditioning = ilu
block 22: Linear System Residual Output = 100
block 22: Linear System Max Iterations = 500
block 22: Linear System Iterative Method = idrs
```

Choice of block strategy

- Using direct method MUMPS for each block we may study the effect of exact block solves on the MLB case
- There really are just two extreme strategies that are useful
  - 1234: block for each velocity + pressure
  - 1112: One block for velocities + pressure
- One velocity block may be reasonable if we find good linear strategy for that
  - In this case scalability better than: 1+log(1.18)/log(3) = 1.15 => Multigrid only!!

Strategy	GCR iters	Cumul. time
1234	766	**29.8**
1112	**648**	63.4
1123	756	37.0
1223	740	37.0
4321	723	32.0
ILU0	1307	29.43
ILU1	723	20.87
ILU2	1297	145.3
MUMPS	NA	29.6

Effect of tolerances

- Block solver utilizes strategies for each block that should be smooth and solved to given precision
- The last decimals of the block solution may be tough to reach
- Relaxing the convergence criteria decreases number of iterations needed drastically
  - may offer great benefits for speed

Strategy	GCR iters	Cumul. time	NRM
1234, e-8	749	35.6	0.90611614
1234, e-7	580	28.7	0.90611614
1234, e-6	421	25.0	0.90611612
1234, e-5	279	19.6	0.90611574
1112, e-8	635	70.3	0.90611614
1112, e-7	491	57.5	0.90611614
1112, e-6	358	45.6	0.90611611
1112, e-5	233	35.4	0.90611569

Block-preconditioner in `IncompressibleNSVec`

Benchmarks `mlb_linsys.sif` on 6 core Intel i5-9400F

- Timings for 50 m resolution case:

Solution strategy	CPU [s]	Real [s]
GCR + ILU 1	27.57	27.98
Block 4 + IDRS	54.81	55.37
Block 4 + BoomerAMG +FlexGMRes & IDRS	123.54	124.05
Block 2 + IDRS	147.34	148.12
Block 2 + BoomerAMG +FlexGMRes & IDRS	444.01	446.60

- Timings for 25 m resolution case:

Solution strategy	CPU [s]	Real [s]
GCR + ILU 1	171.29 /121.49*	172.17 /122.30*
Block 4 + IDRS	310.89/155.41*	311.90/156.26*
Block 4 + BoomerAMG +FlexGMRes & IDRS	520.82	525.09
Block 2 + IDRS	631.43	634.40
Block 2 + BoomerAMG +FlexGMRes & IDRS	1912.73	1919.60

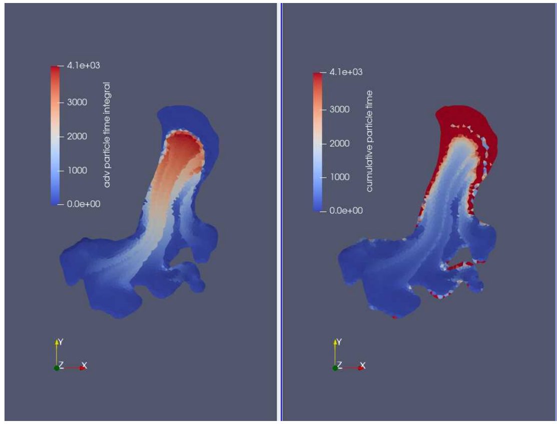
* Reduced tolerance run, change in NRMs 1.5e-6 (ILU) and 1e-8 (block4)

Block4 + idrs revisited

- Comparison of Block4 strategy ILU1 preconditioned strategy
- The sloppier tolerance benefit the block preconditioner much more!
  - 41.4 -> 11.30 s for Block4
  - 30.0 -> 19.0 s for ILU1
- Also the NRM of the nonliear system seems to be less affected
  - 6th vs. 4th digit

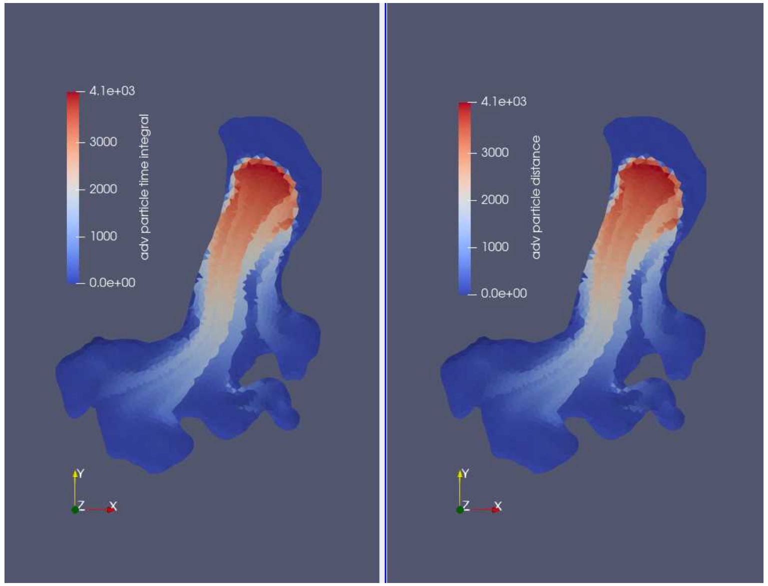
Strategy	GCR iters	Cumul. time	NRM
e-8, e-3	756	41.33	0.90611614
e-7, e-3	570	29.8	0.90611614
e-6, e-3	392	19.6	0.9061161**3**
e-5, e-3	252	12.7	0.906116**2**0
e-5, e-2	264	11.30	0.90611614
e-5, e-1	355	13.73	0.906116**4**3
GCR, e-8	1307	30.0	0.90611606
GCR, e-5	871	19.0	0.906**0**8016

Bug-fix: Semi-Lagrangian example



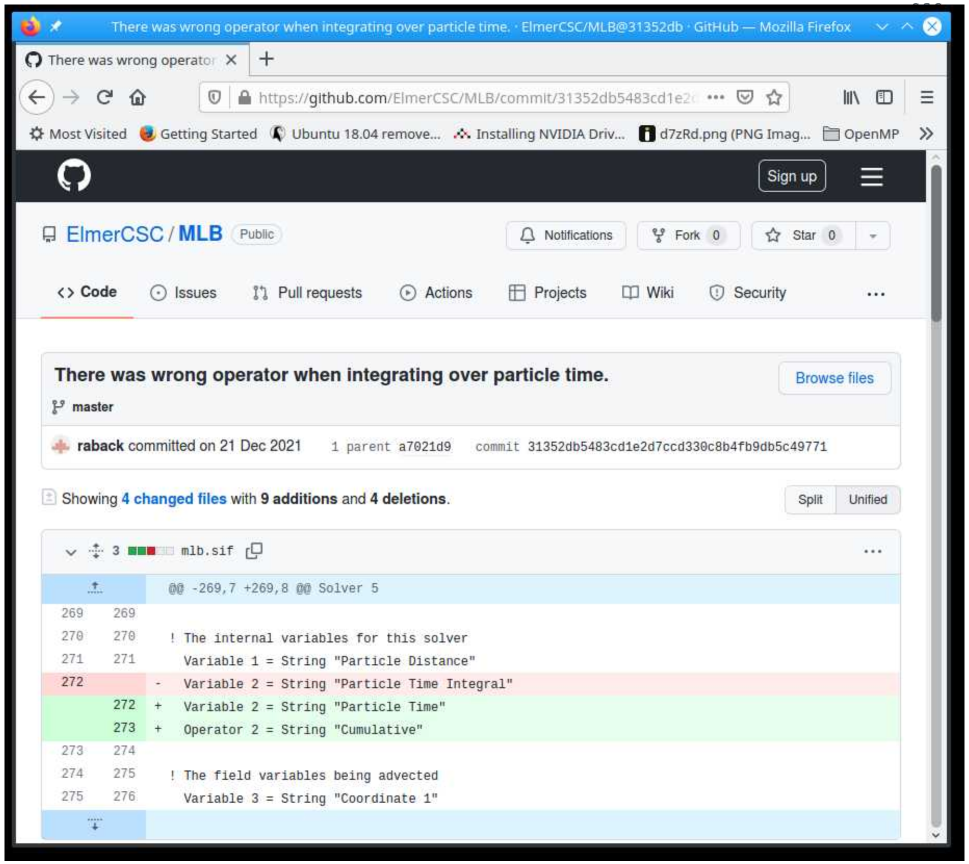
Semi-Lagrangian example

- MLB example had a wrong keyword in the (by default never executed) semi-Lagrangian solver for age/depth evaluation
- One might have realized that the *particle time integral* and the *particle distance* were identical



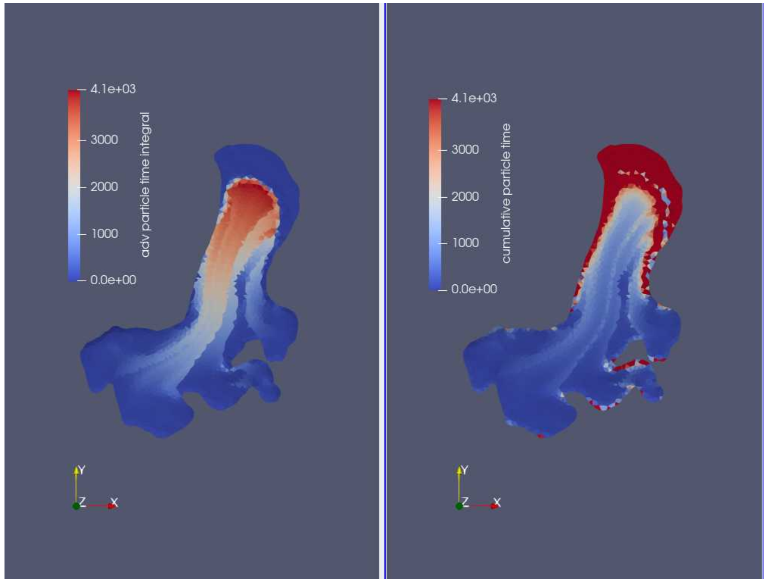
Semi-Lagrangian example

- MLB example had a wrong keyword in the (by default never executed) semi-Lagrangian solver for age/depth evaluation
- One might have realized that the *particle time integral* and the *particle distance* were identical
- Correction: *particle time integral* has been replaced by *particle time* and the operator set to *cumulative*

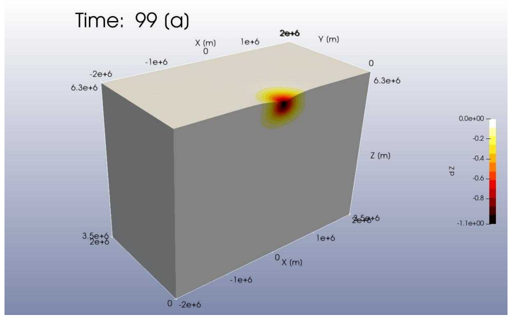


Semi-Lagrangian example

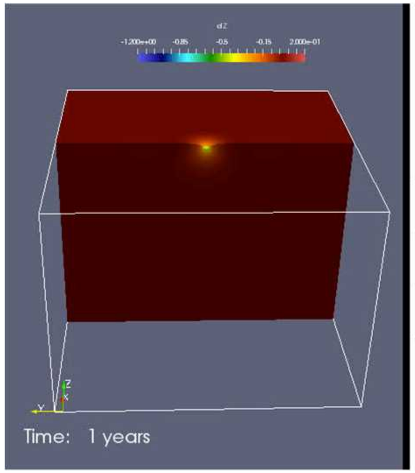
- MLB example had a wrong keyword in the (by default never executed) semi-Lagrangian solver for age/depth evaluation
- One might have realized that the *particle time integral* and the *particle distance* were identical
- Correction: *particle time integral* has been replaced by *particle time* and the operator set to *cumulative*
- Strong reduction of age (except for artefacts in deglaciated areas)

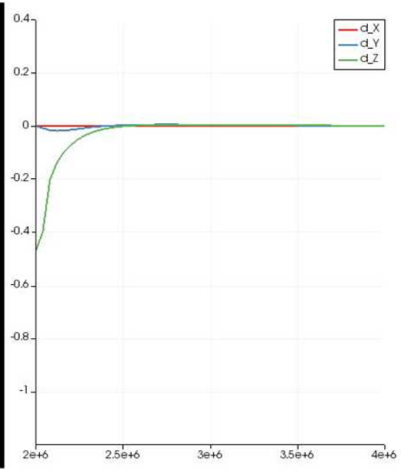


New test-case for Visco-elastic Earth Model



GIA benchmark model





- Visco-elastic – Maxwell rheology :

  (partly non-reversible)

deformation as a function of

viscous and

elastic contribution

Zwinger, T., Nield, G. A., Ruokolainen, J., and King, M. A., 2020. A new open-source viscoelastic solid earth deformation module implemented in Elmer (v8.4), Geosci. Model Dev., 14, 1155–1164, doi:10.5194/gmd-13-1155-2020

Implementation in Elmer

- Introduction of visco-elastic stress (Wu 2004)

$$\frac{\partial \boldsymbol{\tau}}{\partial t} = \frac{\partial \boldsymbol{\tau}_0}{\partial t} + \frac{\mu}{\eta}(\boldsymbol{\tau} - \Pi \mathbf{1})$$

$$\boldsymbol{\tau}_0 = \Pi \mathbf{1} + 2\mu\boldsymbol{\epsilon}$$

  - At the same time we introduce pressure Π to enable incompressibility

(Maxwell time)$^{-1}$

- Additional term accounting for restoring force by specific weight gradient (aka. pre-stress advection)

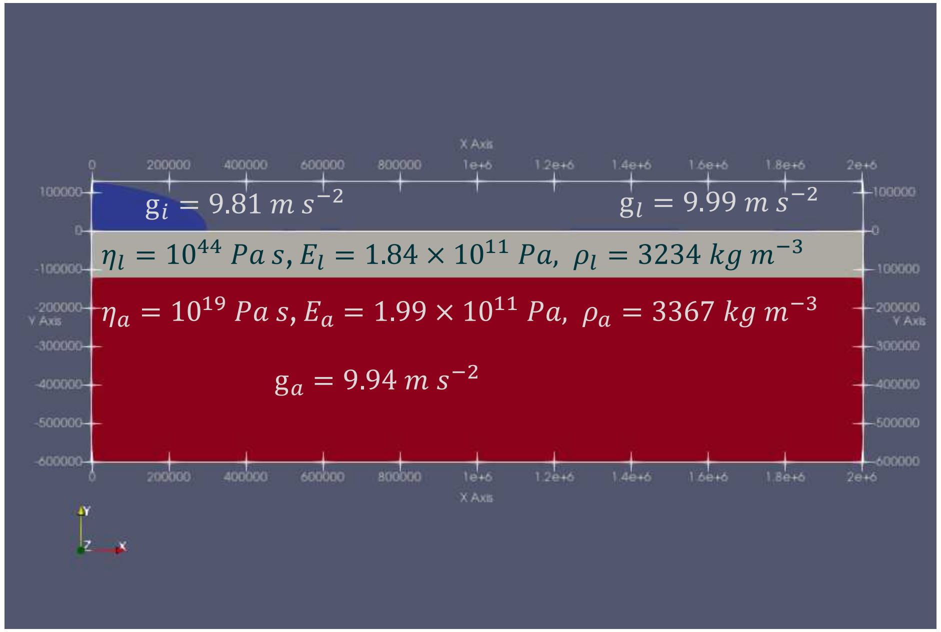
$$\nabla \cdot \boldsymbol{\tau} - \rho g \nabla(\boldsymbol{e}_z \cdot \boldsymbol{d}) = \mathbf{0}$$

  - This is not standard in commercial FE packages, hence needs to be "cheated" around by putting jump-conditions on inter-layer boundaries (Winkler foundations)
  - In Elmer we can include this, which introduces the right boundary condition naturally from the third term of the weak formulation

$$\int_{\Omega} \tau(\boldsymbol{u}) \cdot \boldsymbol{\epsilon}(\boldsymbol{v})\, dV - \oint_{\partial\Omega} (\tau(\boldsymbol{u}) \cdot \boldsymbol{n}) \cdot \boldsymbol{v}\, dA - \int_{\Omega} \rho g \nabla (\boldsymbol{e}_z \cdot \boldsymbol{u}) \cdot \boldsymbol{v}\, dV = 0.$$

Coupled to ice sheet

- Imposing ice sheet with a Bueler profile
- 5 kyr advance from 0-300km (1500 m thickness)
- 5 kyr retreat
- 2 layer model (crust 12 km + mantle 600 km)



Coupled to ice sheet

```
!==============================================================
!---------------- MATERIALS ----------------------
!==============================================================
Material 1
  Name = "Ice Material"
  Density = Real #rhoi
End
! Lithosphere
Material 2
  Density =  #rhol
  Damping = Real 0.0
  Youngs Modulus = #ymodl
  ! supper high viscosity, hence,
  ! Maxwell time is such that it acts elastic
  Viscosity = #viscl
  Poisson Ratio = Real 0.49 !not needed if incompressible
End
! Upper Mantle 1
Material 3
  Density =  #rhoa
  Damping = Real 0.0
  Youngs Modulus = #ymoda
  Viscosity =#visca
  Poisson Ratio = Real 0.49 !not needed if incompressible
End
```

$$\frac{\partial \boldsymbol{\tau}}{\partial t} = \frac{\partial \boldsymbol{\tau}_0}{\partial t} + \frac{\mu}{\eta}(\boldsymbol{\tau} - \Pi \mathbf{1})$$

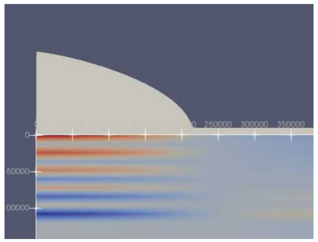
```
!==============================================================
!----------------- BODY FORCES ----------------------
!==============================================================
Body Force 1
  Name = "Ice Bodyforce"
  Flow BodyForce 1 = Real 0
  Flow BodyForce 2 = Real #-gravity
End
! Lithosphere
Body Force 2
 Stress BodyForce 1 =  0.0
 Stress BodyForce 2 =  0.0
 Gravitational Prestress Advection = Logical True
 GPA Coeff = Real # rhol * gravl
End
!Upper Mantle 1
Body Force 3
 Stress BodyForce 1 =  0.0
 Stress BodyForce 2 =  0.0
 Gravitational Prestress Advection = Logical True
 GPA Coeff = Real # rhoa * grava
End
```

$$-\rho g \nabla(e_z \cdot \boldsymbol{d})$$

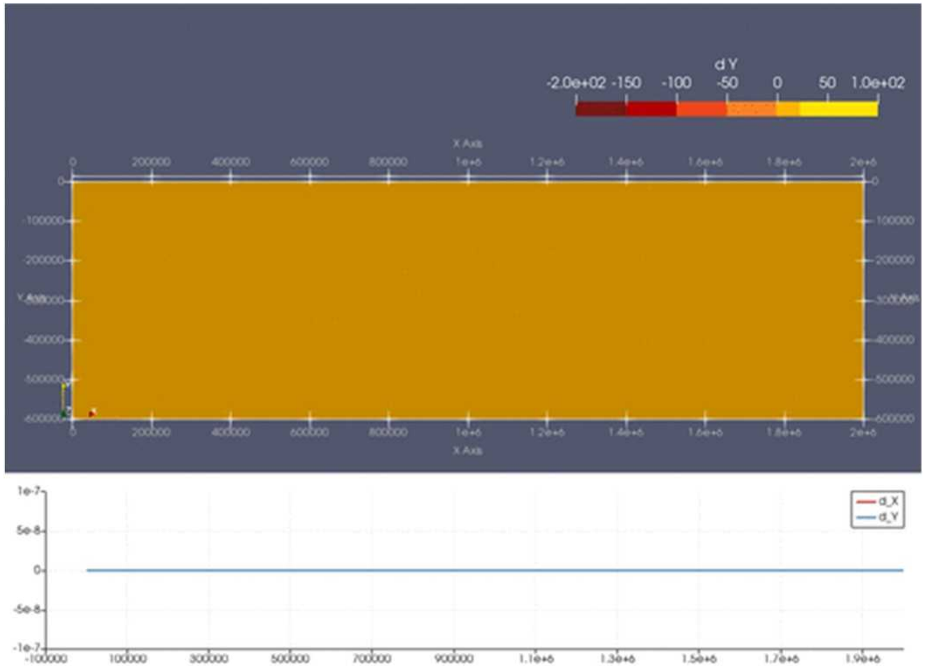
Coupled to ice sheet

```
!=====================================================================
! /// Visco-elastic solver ///
!=====================================================================
Solver 4
  Equation = "Elasticity Analysis"
  Procedure = "StressSolve" "StressSolver"

  Displace Mesh = Logical True ! physically deform the mesh?

  Calculate Stresses = Logical True ! outputs elastic stresses

  ! 2D: 2 deformation 1 pressure (as incompressible)
  Variable = String "t[d:2 p:1]"

  ! if using p:1 and bubble, then either b:3 or b:6
  !  Element = "p:1 b:3 "
  ! best to use p:2 for deofrmation (pressure will be p:1)
  Element = "p:2"

  ! Visco-elastic computation if Ture
  Maxwell material = Logical True
  Incompressible = Logical True
  Time Derivative Order = 1

  ! Numerical settings (here direct Solver)
  Linear System Solver = Direct
  Linear System Direct Method = MUMPS
  Steady State Convergence Tolerance= 1.0e-5
End
```

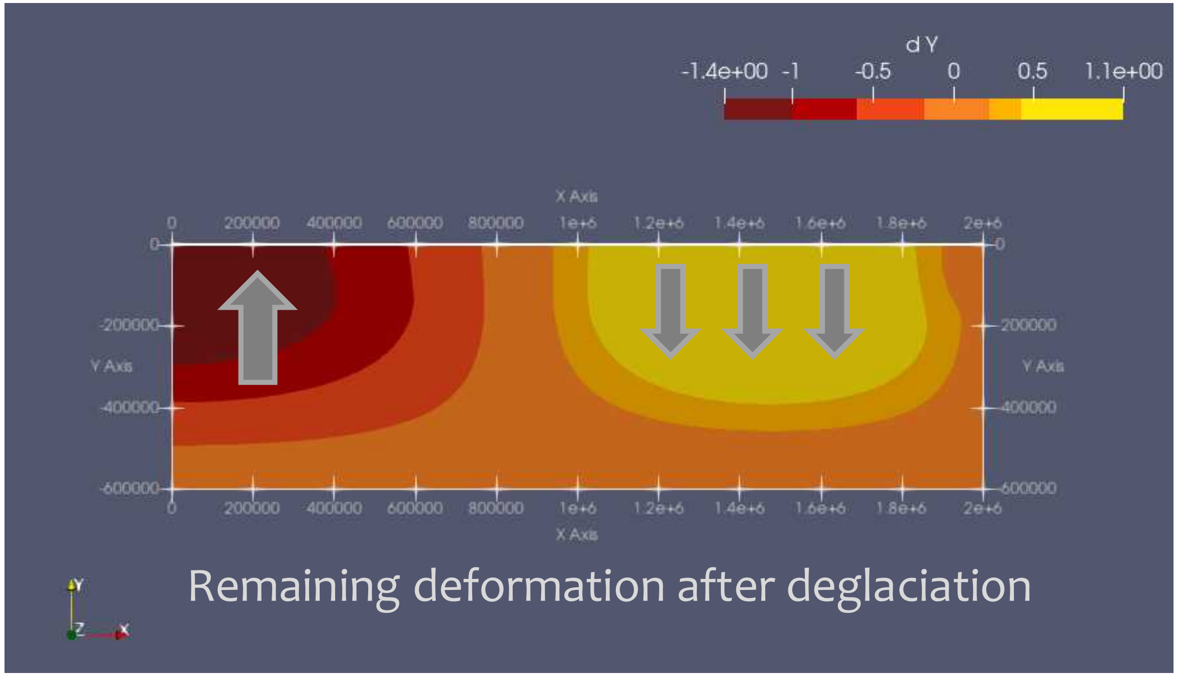
- Add-on functionality to existing linear elasticity solver (`StressSolve`)
- Incompressibility expects deformations + pressure DOF
- Best strategy for stabilization: p:2 (p:1)

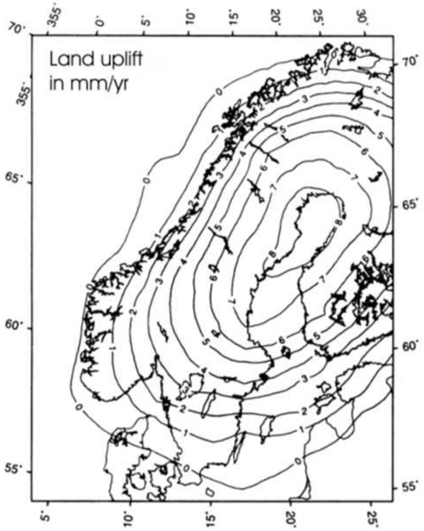


Coupled to ice sheet



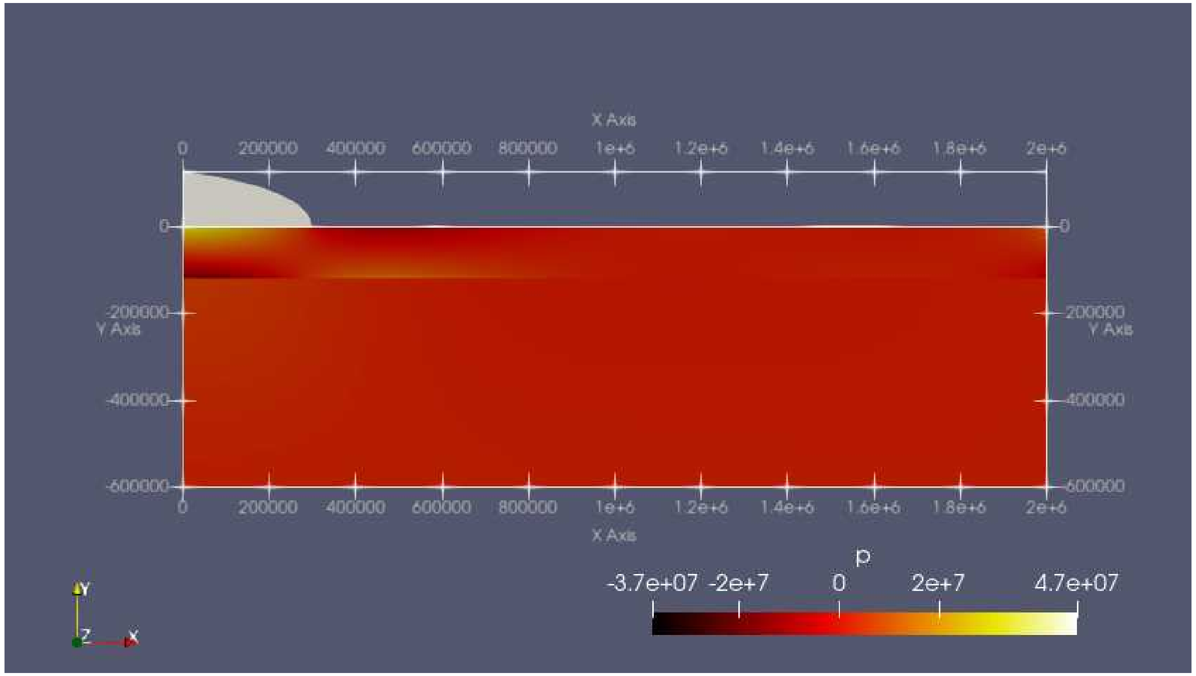
Coupled to ice sheet



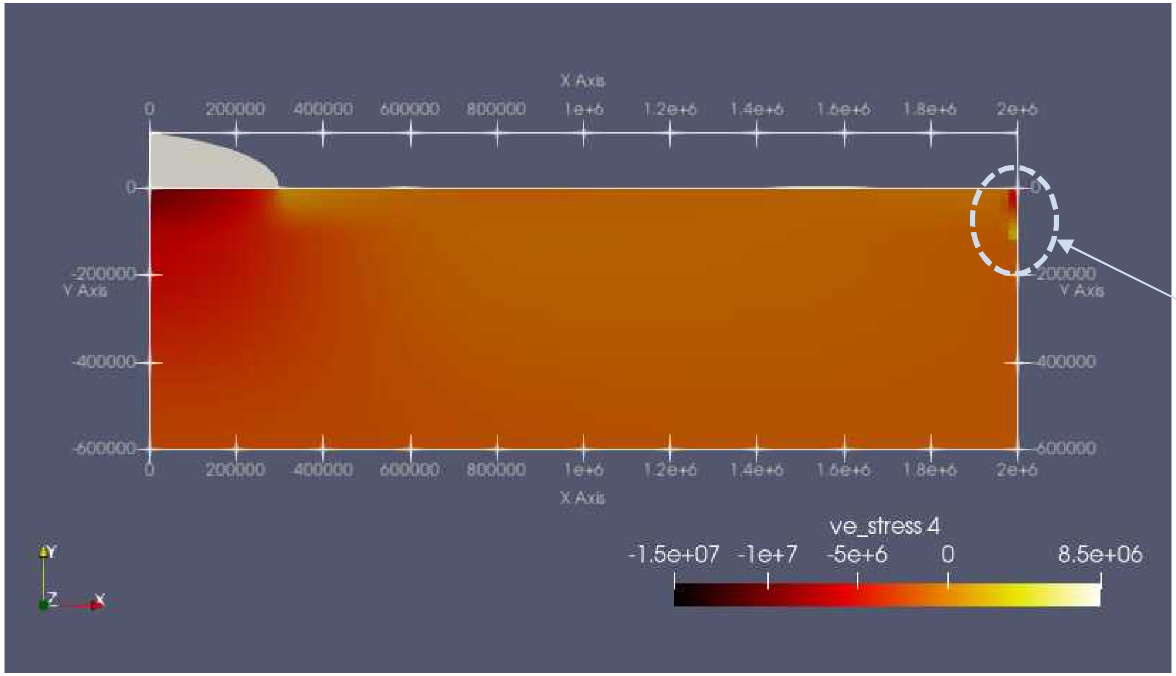


From Thoma and Wolf, 1999

Coupled to ice sheet

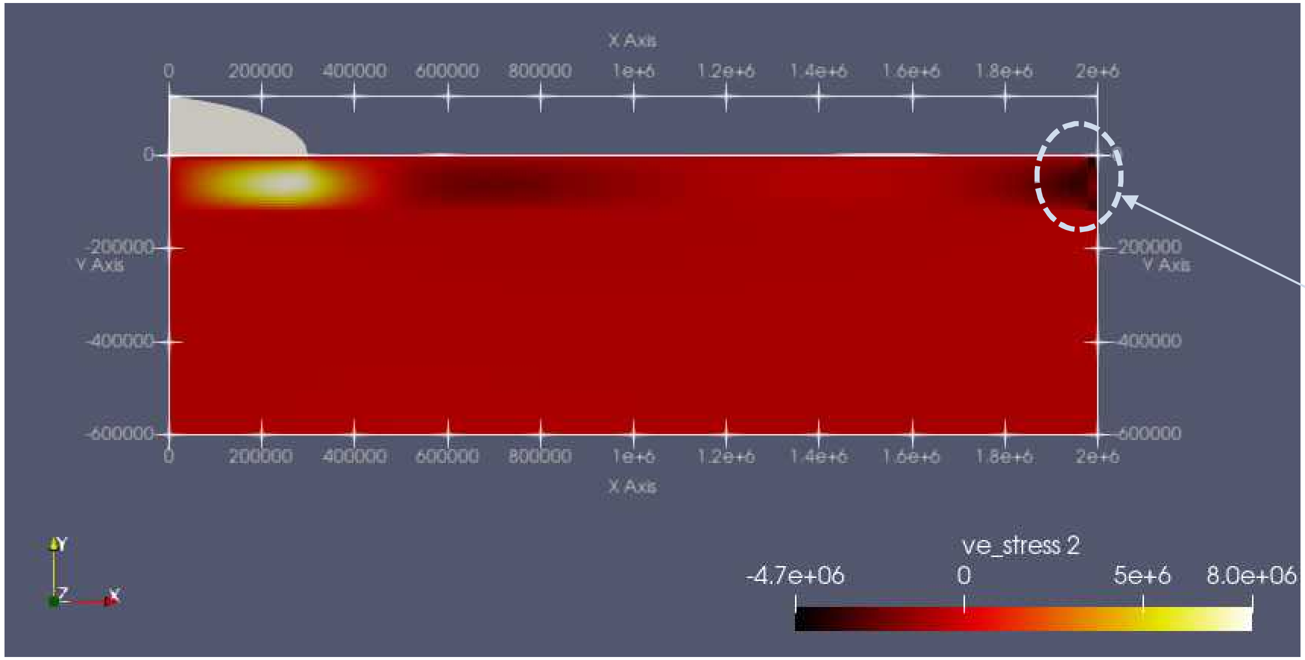


Coupled to ice sheet



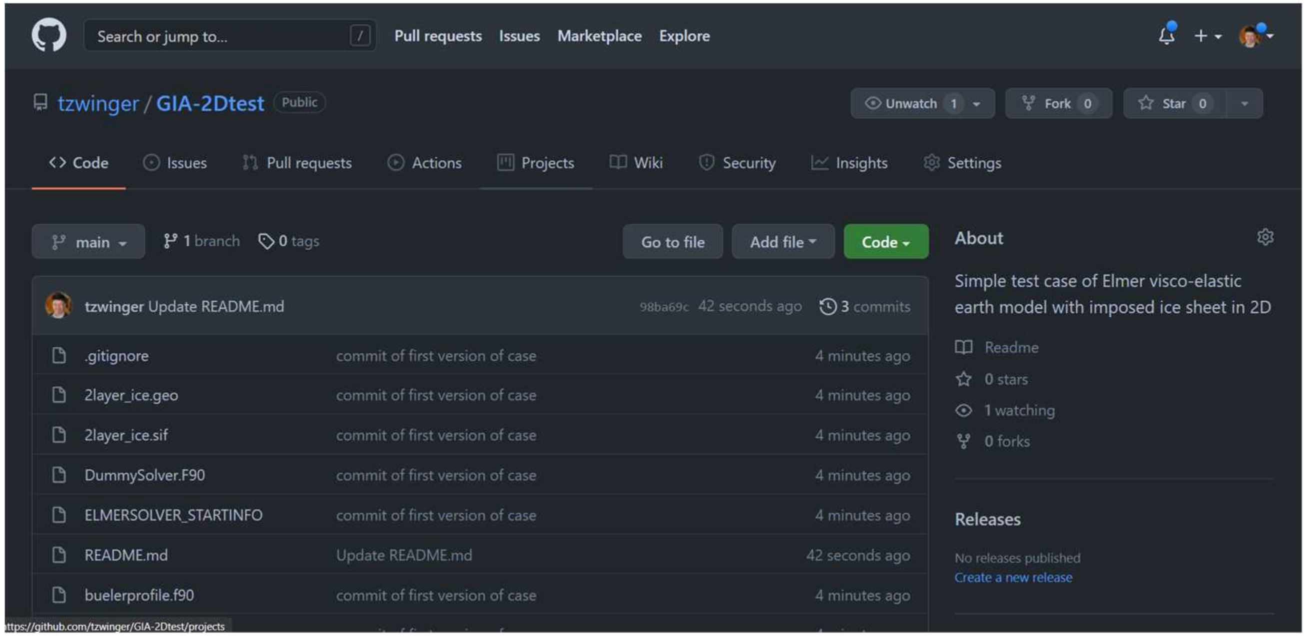
Far-field condition too close

Coupled to ice sheet

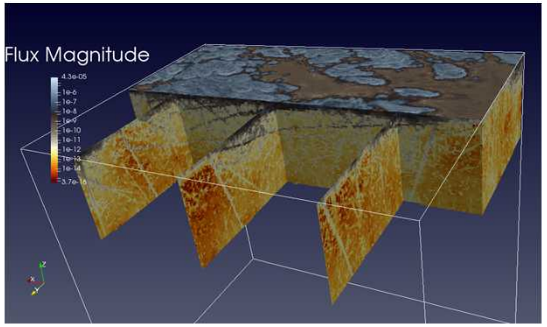


Far-field condition too close

Testcase to be found under GitHub: tzwinger/GIA-2Dtest

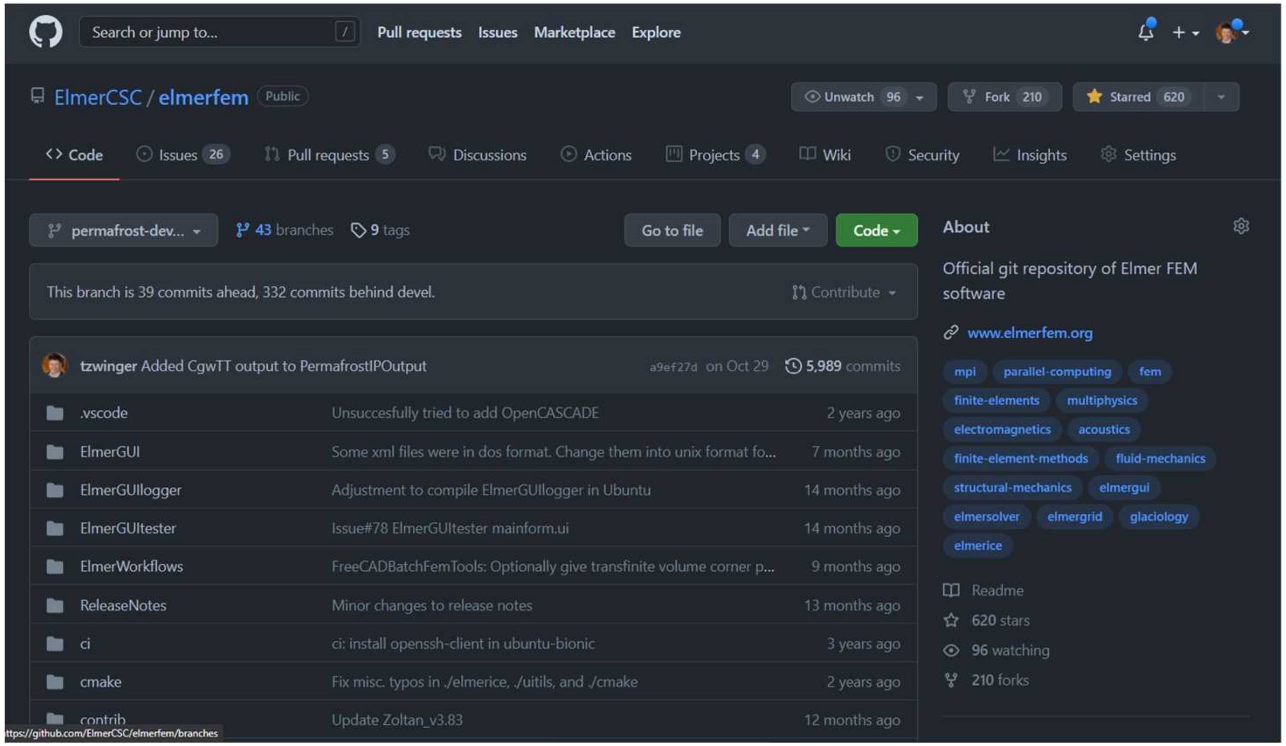


Coupling ice-shet and groundwater-permafrost model

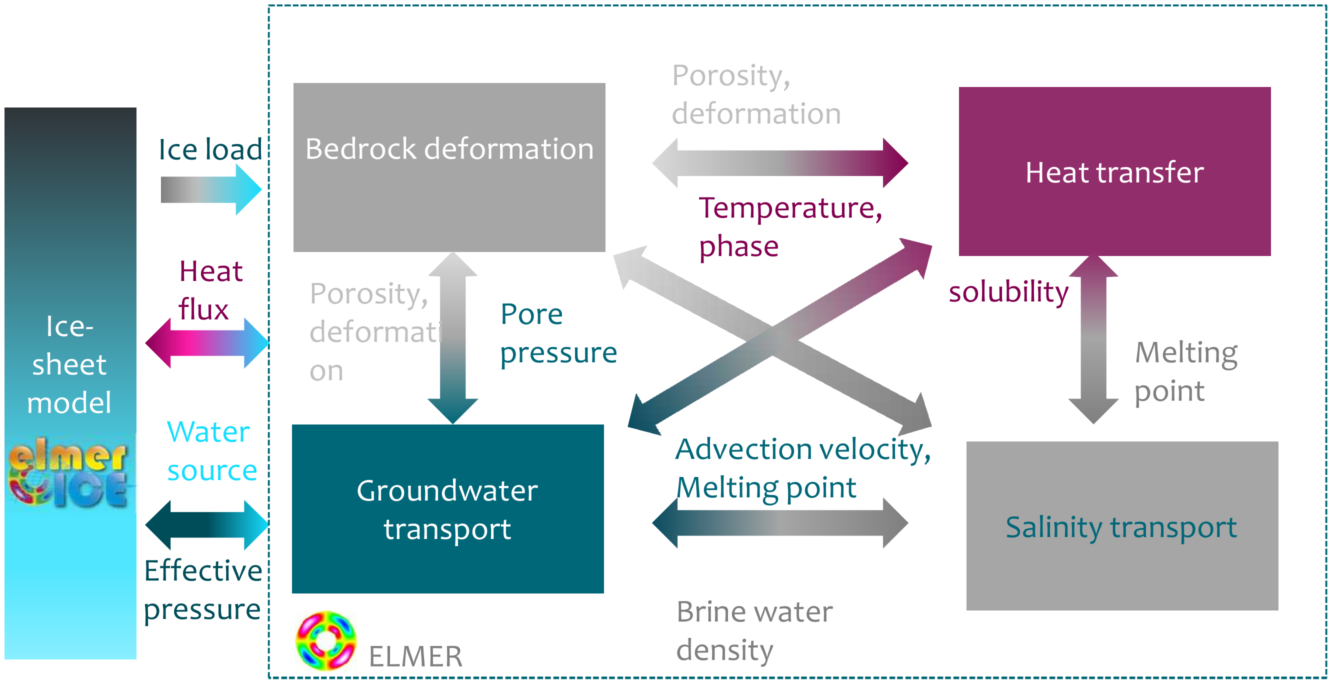


Groundwater-permafrost model

- This is still under heavy development
- Currently, special branch `permafrost-devel` in the GitHub repository is assigned to it

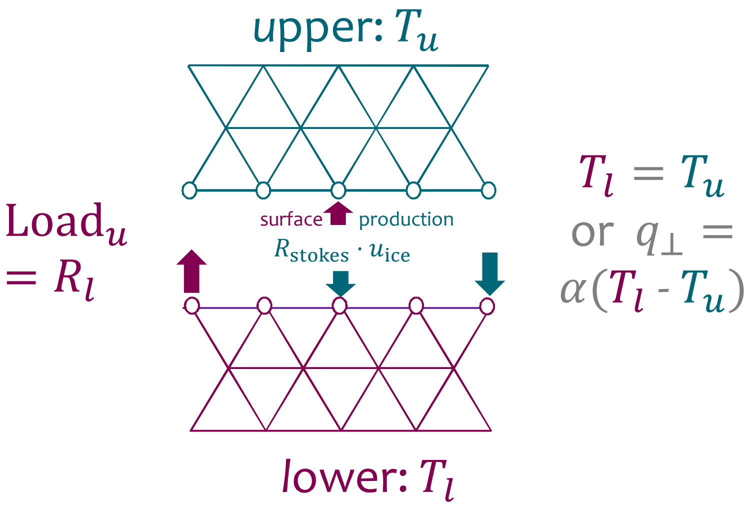


Permafrost model

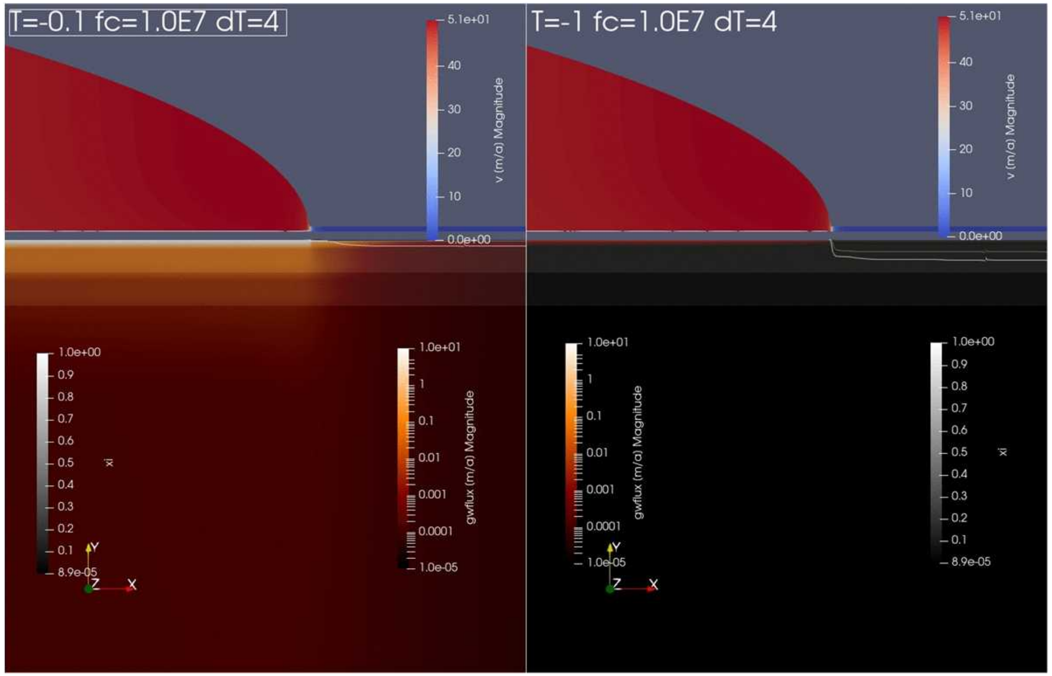


Coupling of ice-sheet to permafrost

- Coupling of solver "of same kind" (e.g. Stokes and lin. Elasticity; HTEQ in ice and permafrost)
- Either Dirichlet-Neumann or Robin-Neumann
- Elegantly using residual as load
- Can also include surface production term



Coupling of ice-sheet to permafrost



CSC

facebook.com/CSCfi

twitter.com/CSCfi

youtube.com/CSCfi

linkedin.com/company/csc---it-center-for-science

github.com/CSCfi

Kuvat CSC:n arkisto, Adobe Stock ja Thinkstock