A real world application
Tête Rousse Glacier

Olivier GAGLIARDINI
LGGE - Grenoble - France
Tête Rousse Glacier

✓ Context
  - The history of Tête Rousse Glacier
  - The 2010 water filled-cavity
  - Analysis of the cavity roof stability (Autumn 2010)

✓ Step 1
  - Tête Rousse Glacier flow without a water filled-cavity (diagnostic)

✓ Step 2
  - Influence of an empty cavity below Tête Rousse Glacier (diagnostic)

✓ Step 3
  - Rate of closure of the cavity for a given drainage scenario (prognostic)
Location (Mont Blanc Area, France Alps)
Location (Mont Blanc Area, France Alps)

Tête Rousse glacier
3100 to 3300 m
0.08 km² (2007)

@B. Jourdain
Chronology

The Past History – The 1892 catastrophe

Contemporary history:

2007-10 - Studies to answer the question about the necessity to maintain the tunnel

07/2010 - A water filled cavity under pressure is discovered
  - Crisis – Artificial drainage

2011 - Small research program to understand the formation of the cavity
  - New crisis – Artificial drainage

2012 - New Artificial drainage needed
The 1892 catastrophe

11 July 1892

175 fatalities

100,000 m$^3$ of water

Flood produced

800,000 m$^3$ of sediment

@Vincent, LGGE
The 1892 catastrophe
Is there still a risk at Tête Rousse?

Question asked by authorities in 2007

@Vincent, LGGE
Glaciological studies

- Topographic measurements
- Radar measurements
- Temperature measurements
- Mass balance measurements
The radar measurements showed a zone (volume) with an anomaly.
Glacialogical studies

In Sept 2009, geophysical survey using the Magnetic Resonance Imaging (LTHE, Grenoble)
Glaciological studies

Water volume of 65 000 m³

Report presented to public authorities in March 2010

@Vincent, LGGE
Pressure measurements

20 hot-water drillings performed from 29 June to 8 July 2010

Confirm the presence of a cavity and that

the cavity is under pressure!
Decisions

The hydrostatic pressure exceeded the ice pressure due to the weight of the ice column

We could expect that the water contained in the glacier would be released suddenly

The public authorities have been warned immediately (13 July, 2010)

It has been decided to drain the subglacial lake as soon as possible, because 3000 people were threatened in the valley.
A difficult field work
Drainage of the cavity

The artificial drainage started the 26 of August
A new risk?

But was stopped on the 1st of September:

What was the risk of collapse of the cavity roof induced by the artificial drainage?
The 2010 cavity

Pumping of 47 700 m$^3$ from 25 August to 8 October 2010

**Question** (addressed end of August 2010):
What is the risk of break-up during the pumping phase?

- right
- upstream
- h $\sim$ 50 m
- l_y $\sim$ 80 m
- l_x $\sim$ 30 m
- left
# Time-line for investigations

<table>
<thead>
<tr>
<th>Date</th>
<th>Event</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>Sonar data</td>
</tr>
<tr>
<td>4, 8, 11</td>
<td>Meeting with the mayor of St Gervais</td>
</tr>
</tbody>
</table>

**Legend:**
- **Sonar data**
- **Meeting with the mayor of St Gervais**
Proposed application

Construct a model of the flow of Tête Rousse Glacier

- Step 1: Without the cavity (normal state)
- Step 2: Add an empty cavity (stress analysis)
- Step 3: Rate of closure of the cavity
  (surface deformation analysis)
Data for ice flow modelling

- Bedrock DEM
- 2007 Surface DEM
- Cavity topography from sonar measurements
- Few surface velocities, without the empty cavity (0.6 m/a at the centre of the glacier)
- 27 Stakes to measure surface displacement during drainage
Material:

Data: Contour_TR_cavity.dat, Contour_TR_glacier.dat, DEM_TR_bed.dat, DEM_TR_cavity.dat, DEM_TR_surf.dat

PROG: USF_TR.f90

Step1: Makegeo.m, teterousse.geo, teterousse1.sif

Step2a: Makegeo_2.m, teterousse.geo, teterousse2a.sif

Step2b: teterousse2b.sif

Step3a: teterousse3a.sif

Step3b: teterousse3b.sif
Modelling Tête Rousse Glacier

✓ **Step 1** - Tête Rousse Glacier flow without a water filled-cavity (diagnostic)

✓ **Step 2**
- 2a Influence of an empty cavity below Tête Rousse Glacier (diagnostic)
- 2b Apply a water pressure in the cavity

✓ **Step 3**
- 3a Rate of closure of the cavity for a given drainage scenario (prognostic)
- 3b Add a drainage scenario
Step 1: Work to do

- create the mesh
- impose the boundary conditions in the SIF file
- test other BCs on the lateral boundary
- test sliding at the base of the glacier
Step 1: steps to make the mesh

1/ build the **teterousse.geo** file (input file of gmsh, footprint of the glacier)

2/ **gmsh** to get **teterousse.msh** (still footprint of the glacier)

3/ **ElmerGrid** to transform into Elmer format (still footprint of the glacier)

4/ we will use the internal extrusion feature in Elmer to create a volume from this footprint (see the sif file)
clear;
lc_out=18.0;  \hspace{1cm} \text{(size of the element in the plane)}

A=dlmread('Contour_TR_glacier.dat'); \hspace{1cm} \text{(Read contour points)}
fid1=fopen('teterousse.geo','w');
fprintf(fid1,'Mesh.Algorithm=5; \n'); \hspace{1cm} \text{(delaunay algorithm)}

As=size(A,1);
np=0;
for ii=1:As
    np=np+1;
    fprintf(fid1,'Point(%g)={%14.7e,%14.7e,0.0,%g}; \n',np,A(ii,1),A(ii,2),lc_out);
end

fprintf(fid1,'Spline(1)=
');
for ii=1:As
    fprintf(fid1,'%g,',ii);
end
fprintf(fid1,'%g}; \n',1);

fprintf(fid1,'Line Loop(2)={1}; \n');
fprintf(fid1,'Plane Surface(3) = {2}; \n');
fprintf(fid1,'Physical Line(4) = {1}; \n');
fprintf(fid1,'Physical Surface(5) = {3}; \n');

fclose(fid1)
Step 1a: Makegeo.m

% create teterousse.msh using gmsh
system "gmsh teterousse.geo -2"

% convert teterousse.gmsh in an Elmer type mesh
System "ElmerGrid 14 2 teterousse.msh -autoclean"
Step 1: \texttt{gmsh} (create a .msh file)

\begin{verbatim}
gmsh teterousse.geo -2
\end{verbatim}

help: \url{http://www.geuz.org/gmsh/}

line commands:
"-2" performs 1D and 2D mesh generation and then exit
Step 1: In the sif file

Define the number of vertical layers (Simulation section):

Simulation
  Coordinate System = Cartesian 3D
  Simulation Type = Steady
  Extruded Mesh Levels = Integer 16

... Integer 16

End

The second solver to be executed is the StructuredMeshMapper

Solver 2
  Equation = "MapCoordinate"
  Procedure = "StructuredMeshMapper" "StructuredMeshMapper"
  Active Coordinate = Integer 3 (3d problem - mesh moves in z direction)
  Mesh Velocity Variable = String "dSdt"
  Mesh Update Variable = String "dS"
  Mesh Velocity First Zero = Logical True
  Displacement Mode = Logical False
  Correct Surface = Logical True
  Minimum Height = Real 1.0

zs = min(zs, bed+1.0)
Step 1: In the sif file

Read, interpolate and store in 2 variables the bed and surface DEM

Solver 1

  Exec Solver = "Before Simulation"
  Equation = "Read DEMs"
  Procedure = "ElmerIceSolvers" "Grid2DInterpolator"

  ! Bedrock DEM
  Variable 1 = String "bedDEM"
  Variable 1 data file = File "../Data/DEM_TR_bed.dat"  name of the DEM file
  Variable 1 x0 = Real 947700.0d0
  Variable 1 y0 = Real 2104850.0d0
  Variable 1 lx = Real 600.0
  Variable 1 ly = Real 350.0
  Variable 1 Nx = Integer 301
  Variable 1 Ny = Integer 176
  Variable 1 Invert = Logical False
  Variable 1 Fill = Logical False
  Variable 1 Position Tol = Real 1.0e-1
  Variable 1 No Data = Real -9999.0
  Variable 1 No Data Tol = Real 1.0

  ! Surface DEM ...

End

define the DEM file structure

(name of the DEM file)

\( (x_0, y_0) \)

\( l_x \)  \( l_y \)

\( N_x + 1 \) columns

\( N_y + 1 \) lines
Step 1: In the sif

BedDEM and ZsDEM (variable) must be declared in one solver (Stokes for example)

Exported Variable 3 = -dofs 1 "BedDEM"
Exported Variable 4 = -dofs 1 "ZsDEM"

Keywords Bottom Surface and Top Surface (needed by the solver StructuredMeshMapper) are assigned the value of these two variables

!Bed rock BC
Boundary Condition 2
  Bottom Surface = Equals BedDEM

...

End

! Upper Surface BC
Boundary Condition 3
  Top Surface = Equals ZsDEM
End
Step 1: use Glen’s law

\[ D_{ij} = \frac{A}{\tau_e^{n-1}} S_{ij} \quad ; \quad S_{ij} = \frac{A^{-1/n} I_{D_2}^{(1-n)/n}}{D_{ij}} \]

\[ A = A(T') = A_0 \exp^{-Q/RT'} \]

\[ A = A_1 = 2.89 \times 10^{-13} \, \text{s}^{-1} \text{Pa}^{-3} \text{ if } T \leq -10^\circ \text{C} \]
\[ A = A_2 = 2.43 \times 10^{-2} \, \text{s}^{-1} \text{Pa}^{-3} \text{ if } T \geq -10^\circ \text{C} \]

\[ Q = Q_1 = 60 \, \text{kJ mol}^{-1} \text{ if } T \leq -10^\circ \text{C} \]
\[ Q = Q_2 = 115 \, \text{kJ mol}^{-1} \text{ if } T \geq -10^\circ \text{C} \]

Cuffey and Paterson (2010)

assume a constant temperature of -1°C
Step 1: use Glen’s law

\[ \text{yearinsec} = 365.25\times24\times60\times60 \]
\[ \rho_{\text{hi}} = \frac{900.0}{1.0\times10^6\times\text{yearinsec}^2} \]
\[ \rho_{\text{h}} = \frac{1000.0}{1.0\times10^6\times\text{yearinsec}^2} \]
\[ \text{gravity} = -9.81\times\text{yearinsec}^2 \]

! Prefactor from Cuffey & Paterson (2010) in MPa^{-3} a^{-1}

\[ A_1 = 2.89165\times10^{-13}\times\text{yearinsec}\times1.0\times10^{18} \]
\[ A_2 = 2.42736\times10^{-2}\times\text{yearinsec}\times1.0\times10^{18} \]
\[ Q_1 = 60.0\times10^3 \]
\[ Q_2 = 115.0\times10^3 \]

Material 1
- Density = Real \( \rho_{\text{hi}} \)
- Viscosity Model = String "glen"
- Viscosity = 1.0 ! Dummy but avoid warning output
- Glen Exponent = Real 3.0
- Limit Temperature = Real -10.0
- Rate Factor 1 = Real \( A_1 \)
- Rate Factor 2 = Real \( A_2 \)
- Activation Energy 1 = Real \( Q_1 \)
- Activation Energy 2 = Real \( Q_2 \)
- Glen Enhancement Factor = Real 1.0
- Critical Shear Rate = Real 1.0\times10^{-10}

Constant Temperature = Real -1.0

End
Step 1: Hypothesis of the modelling

Solve only the Stokes equation in a diagnostic way

3 boundary conditions

Bedrock BC:
No sliding condition

Upper Surface BC:
stress free surface
(natural BC)

Lateral BC:
zero horizontal velocities
Step 1: Boundary Conditions

! lateral side of the glacier
Boundary Condition 1
  Target Boundaries = 1
  Velocity 1 = real 0.0
  Velocity 2 = real 0.0
End

! Bedrock
Boundary Condition 2
  Bottom Surface = Equals BedDEM
  Velocity 1 = Real 0.0
  Velocity 2 = Real 0.0
  Velocity 3 = Real 0.0
End

! Upper Surface
Boundary Condition 3
  Top Surface = Equals ZsDEM
End
Step 1: Stress Solver

Objective: compute the stress field as

\[ \int_V S_{ij} \Phi \, dV = 2 \int_V \eta D_{ij} \Phi \, dV \]

where \( D_{ij} \) and \( \eta \) are calculated from the nodal velocities using the derivative of the basis functions.

Solver 4

Equation = Sij
Procedure = "ElmerIceSolvers" "ComputeDevStress"
Variable = -nooutput "Sij"
Variable DOFs = 1
Exported Variable 1 = Stress
Exported Variable 1 DOFs = 6

Flow Solver Name = String "Flow Solution"

Linear System Solver = Direct
Linear System Direct Method = umfpack
End
Step 1: Stress Solver

- Tell you want the Cauchy stress to be computed (Material Section) (else you will get the deviatoric stress)

Material 1
Cauchy Stress = Logical True
End

- Output:
  negative stress = Compressive stress
  positive stress = Tensile stress

\[
\begin{align*}
\text{Stress.1} & \rightarrow S_{xx} \\
\text{Stress.2} & \rightarrow S_{yy} \\
\text{Stress.3} & \rightarrow S_{zz} \\
\text{Stress.4} & \rightarrow S_{xy} \\
\text{Stress.5} & \rightarrow S_{yz} \\
\text{Stress.6} & \rightarrow S_{xz}
\end{align*}
\]
Step 1: Eigenvalues Solver

Objective: compute the eigenvalues of the Cauchy stress tensor

Solver 5
  Equation = "EigenStresses"
  Procedure = "ElmerIceSolvers" "ComputeEigenValues"
  Variable = -nooutput dumy
  Variable DOFs = 1

! The 3 eigenvalues
  Exported Variable 1 = EigenStress
  Exported Variable 1 DOFS = 3

! The 3 eigenvectors (Option)
  Exported Variable 2 = EigenVector1
  Exported Variable 2 DOFS = 3
  Exported Variable 3 = EigenVector2
  Exported Variable 3 DOFS = 3
  Exported Variable 4 = EigenVector3
  Exported Variable 4 DOFS = 3

End
Step 1: Eigenvalues Solver

Output:

- negative stress = Compressive stress
- positive stress = Tensile stress
- ordered → Eigenstress.3 gives the maximal tensile stress

Eigenstress.1 → $S_1$
Eigenstress.2 → $S_2$
Eigenstress.3 → $S_3$
Step 1: Add sliding on the bedrock

Friction law in Elmer:

\[ C_i u_i = \sigma_{ij} n_j \quad (i = 1, 2) \]

\[ c_t u_t = \sigma_{nt} \quad ; \quad c_n u_n = \sigma_{nn} \]

where \( n \) is the surface normal vector

! Bedrock BC  
Boundary Condition 2

Flow Force BC = Logical True  
Normal-Tangential Velocity = Logical True

Velocity 1 = Real 0.0e0  
Slip Coefficient 2 = Real 0.1  
Slip Coefficient 3 = Real 0.1

End

How to evaluate the Slip Coefficient?
Step 1: Other BCs for the lateral boundary

! lateral side of the glacier
Boundary Condition 1
   Target Boundaries = 1
End

Natural BC

! lateral side of the glacier
Boundary Condition 1
   Target Boundaries = 1
   Velocity 1 = real 0.0
   Velocity 2 = real 0.0
   Velocity 3 = real 0.0
End

zero velocity

Conclusion ?
Modelling Tête Rousse Glacier

✓ Step 1 - Tête Rousse Glacier flow without a water filled-cavity (diagnostic)

✓ Step 2
- 2a Influence of an empty cavity below Tête Rousse Glacier (diagnostic)
- 2b Apply a water pressure in the cavity

✓ Step 3
- 3a Rate of closure of the cavity for a given drainage scenario (prognostic)
- 3b Add a drainage scenario
Step 2a: Add the cavity (empty of water)

The initial problem

The new problem

Two BCs for the base:

\[ u = 0 \text{ if } z_b = b \]
\[ \sigma_{nn} = p_w \text{ if } z_b > b \]
\[ p_w = 0 \text{ if the cavity is empty of water} \]

!!! The ice bottom surface is not anymore given by the bedrock DEM !!!
Step 2a: new Bottom Surface definition

The bottom surface is now given by the DEM_TR_cavity.dat DEM file.

Change the StructuredMeshMapper to read this DEM and store it in the ZbDEM variable

Declare this new variable (in Stokes solver section):
Exported Variable 5 = -dofs 1 "ZbDEM"

Change the boundary condition 2:
Boundary Condition 2
  Bottom Surface = Equals ZbDEM
End
Step 2a: Make a new mesh

We will use the cavity contour to have smaller size elements in the vicinity of the cavity

**Work to do**: modify the `Makegeo.m` file to create this new mesh.
Step 2a: Make a new mesh (Makegeo_2.m) 1/2

clear;
lc_out=18.0;
lc_in=6.0;
A=dlmread('Contour_TR_glacier.dat');
B=dlmread('Contour_TR_cavity.dat');
fid1=fopen('teterousse.geo','w');
As=size(A,1);
Bs=size(B,1);
np=0;
for ii=1:As
    np=np+1;
    fprintf(fid1,'Point(%g)={%14.7e,%14.7e,0.0,%g}; \n',np,A(ii,1),A(ii,2),lc_out);
end
for ii=1:Bs
    np=np+1;
    fprintf(fid1,'Point(%g)={%14.7e,%14.7e,0.0,%g}; \n',np,B(ii,1),B(ii,2),lc_in);
end
fprintf(fid1,'Spline(1)={
        for ii=1:As
            fprintf(fid1,'%g,',ii);
        end
        fprintf(fid1,'};
    end
    fprintf(fid1,'Spline(2)={
        for ii=1:Bs
            fprintf(fid1,'%g,',As+ii);
        end
        fprintf(fid1,'};
    end

Step 2a: Make a new mesh (Makegeo_2.m) 2/2

```matlab
fprintf(fid1,'Line Loop(3)={1}; \n');
fprintf(fid1,'Line Loop(4) = {2}; \n');
fprintf(fid1,'Plane Surface(5) = {3, 4}; \n');
fprintf(fid1,'Plane Surface(6) = {4}; \n');
fprintf(fid1,'Physical Line(7) = {1}; \n');
fprintf(fid1,'Physical Surface(8) = {5,6}; \n');
fclose(fid1)

% create teterousse.msh using gmsh
system "gmsh teterousse.geo -1 -2"

% convert teterousse.gmsh in an Elmer type mesh
System "ElmerGrid 14 2 teterousse.msh -autoclean"
```
Step 2a: Change in the basal BC

The basal BC will be of the form:

Velocity 1 = Real 0.0
Velocity 1 Condition = Variable ZbDEM, BedDEM
Real MATC "-(tx(0) > tx(1))"

Velocity 1 Condition = -1 if zbDEM > BedDEM => Don’t apply “Velocity 1 = 0”
= +1 if zbDEM <= BedDEM => Apply “Velocity 1 = 0”

And the same for Velocity 2 and Velocity 3.

Visualize the results in ElmerPost.
What does it change in term of velocity and stress?
Step 2a: Change in the basal BC

2nd Solution – use a f90 user function instead of MATC language

The basal BC will be of the form:

- Velocity 1 = Real 0.0
- Velocity 1 Condition = Variable ZbDEM, BedDEM
- Real Procedure “../PROG/USF_TR” “MaskCavity”

Compile the user function USF_TR.f90 in PROG/

elmerf90 USF_TR.f90 -o USF_TR

Best solution for large problem:
- a f90 user function is much faster than MATC coding in the sif!
Step 2a: MaskCavity user function

FUNCTION MaskCavity ( Model, nodenumber, Input) RESULT(Mask) 
USE types 
USE CoordinateSystems 
USE SolverUtils 
USE ElementDescription 
USE DefUtils 
IMPLICIT NONE 
TYPE(Model_t) :: Model 
TYPE(Solver_t), TARGET :: Solver 
INTEGER :: nodenumber 
REAL(KIND=dp) :: Input(2), Mask 
REAL(KIND=dp) :: znode, zbed 

znode = Input(1) 
zbed = Input(2) 

IF (znode > zbed+0.1) THEN 
 Mask = -1.0 
ELSE 
 Mask = 1.0 
END IF 
END FUNCTION MaskCavity
Modelling Tête Rousse Glacier

- **Step 1** - Tête Rousse Glacier flow without a water filled-cavity (diagnostic)

- **Step 2**
  - 2a Influence of an empty cavity below Tête Rousse Glacier (diagnostic)
  - 2b Apply a water pressure in the cavity

- **Step 3**
  - 3a Rate of closure of the cavity for a given drainage scenario (prognostic)
  - 3b Add a drainage scenario
Step 2b: Add a water pressure

Modify the SIF to add a water pressure

\[ p_w = \rho_w g (h_w - z) \]

water load \( h_w \)

\[ \text{the water load} \]

$\text{hw} = 3176.0$

In the bedrock BC

Flow Force BC = Logical True
External Pressure = Variable Coordinate 3
Real MATC “rhow*gravity*(hw-tx)"

will only apply where a Dirichlet BC is not applied i.e. in the cavity
Modelling Tête Rousse Glacier

- **Step 1** - Tête Rousse Glacier flow without a water filled-cavity (diagnostic)

- **Step 2**
  - 2a Influence of an empty cavity below Tête Rousse Glacier (diagnostic)
  - 2b Apply a water pressure in the cavity

- **Step 3**
  - 3a Rate of closure of the cavity for a given drainage scenario (prognostic)
  - 3b Add a drainage scenario
Step 3a: Move to prognostic

Will do it in two steps
- Move to prognostic assuming the cavity is empty of water at t=0 (big step, need 2 new solvers!)
- Prescribe the observed drainage scenario for the water pressure

To move from a diagnostic to a prognostic simulations:
- Add the FreeSurface solver (here 2 times, since we have 2 FS)
- Add one body per FS (new Initial Condition and Equation Sections)
- Modifications in the Simulation and Boundary Condition Sections

Only shown for the upper free surface here
The simulation Section has to be modified:

Simulation Type = Transient
Timestepping Method = “bdf” → Backward Differences Formulae
BDF Order = 1
Output Intervals = 1
Timestep Intervals = 50
Timestep Sizes = $10.0/365.25$

Steady State Min Iterations = 1
Steady State Max Iterations = 1

Restart File = "../..//Step2a/teterousse/teterousse_Step2a_.result"
Restart Position = 0
Restart Time = Real 0.0
Restart Before Initial Conditions = Logical True

We need a restart to have the ZsDEM and ZbDEM variables for the initial condition of Zs and Zb

To control the “implicity” of the solution over one time step (here 1 means explicit)
Step 3a – Sketch of a transient simulation

Geometry + Mesh → Degrees of freedom

Linear System: $A \cdot x = b$

- Direct
- Iterative $\epsilon_L$

Non Linear iterations: $A = A(x)$

Solver $k$

Coupled iteration over $dt$ (implicit scheme)

$t = t + dt$

$\epsilon_L < \epsilon_{NL} < \epsilon_C$
Step 3a – Free surface Solver

The free surface solver only apply to the boundary 3 (upper surface)
→ Define a 2nd body which is the boundary 3.

Body 2
  Equation = 2
  Body Force = 2
  Material = 1
  Initial Condition = 2
End

where Equation 2, Body Force 2 and Initial Condition 2 are defined for the free surface equation of the upper surface.

Tell in BC2 that this is the body 2:
  Boundary Condition 3
    Body Id = 2
    ...
End
Step 3a – Add the Free surface Solver

Solver 4
  Equation = "Free Surface Top"
  Variable = String "Zs"
  Variable DOFs = 1
  Exported Variable 1 = String "Zs Residual"
  Exported Variable 1 DOFs = 1

  Procedure = "FreeSurfaceSolver" "FreeSurfaceSolver"
  Before Linsolve = "EliminateDirichlet" "EliminateDirichlet"

  Linear System Solver = Iterative
  Linear System Max Iterations = 1500
  Linear System Iterative Method = BiCGStab
  Linear System Preconditioning = ILU0
  Linear System Convergence Tolerance = Real 1.0e-9
  Linear System Abort Not Converged = False
  Linear System Residual Output = 1

  Nonlinear System Max Iterations = 100
  Nonlinear System Convergence Tolerance = 1.0e-6
  Nonlinear System Relaxation Factor = 1.00

  Steady State Convergence Tolerance = 1.0e-03

  Stabilization Method = Bubbles
  Apply Dirichlet = Logical False

  ! How much the free surface is relaxed
  Relaxation Factor = Real 1.00

End
Step 3a – Upper Surface

Body Force 2:

Body Force 2
  Zs Accumulation Flux 1 = Real 0.0e0
  Zs Accumulation Flux 2 = Real 0.0e0
  Zs Accumulation Flux 3 = Real 0.0e0
End

Equation 2:

Equation 2
  Active Solvers(1) = 2
  Flow Solution Name = String “Flow Solution”
  Convection = String Computed
End

Initial Condition 2: (tell that $z_s(x, 0)$ is given by the surface DEM)

Initial Condition 2
  Zs = Equals ZsDEM
End
Step 3a - StructuredMeshMapper

We say in StructuredMeshMapper that the top (and bottom) surface is defined by the variable zs:

Solver 1
   Equation = "MapCoordinate"
   Procedure = "StructuredMeshMapper" "StructuredMeshMapper"

   Active Coordinate = Integer 3
   Mesh Velocity Variable = String "dSdt"
   Mesh Update Variable = String "dS"
   Mesh Velocity First Zero = Logical True

   Top Surface Variable Name = String "Zs"
   Bottom Surface Variable Name = String "Zb"

End

And delete from the BC the initial definition of the top (and bottom) surface:

Boundary Condition 3
   !!! this BC is equal to body no. 2 !!!
   Body Id = 2
   Top Surface = Equals ZsDEM

End
Step 3a – Same for the bedrock

Name of the variable: Zs Bottom
Add solver: Solver 5
Add equation: Equation 3

Modify the the Bottom surface BC (3):

```
Boundary Condition 2
  Body Id = 3
  Bottom Surface = Equals ZbDEM
End
```

Add a limiter to ensure that $z_b \geq b$
In the material section

```
Min Zb = Equals BedDEM
Max Zb = Real +1.0e10
```

+ in the Free Surface solver: Apply Dirichlet = Logical True
Step 3a – Newton linearization

If you want to use Newton linearization for the non-linear iterations, don’t forget to reset the conditions used to move from Picard to Newton at each time step, by adding:

Solver 1

    Nonlinear System Reset Newton = Logical True

End
Step 3a – Two subtleties…

1/ Need of the restart...
   Initial conditions are set before the first solver is executed
   Impossible then to initialize with an other variable
   This is then done by using a restart and specifying:

   `Restart Before Initial Conditions = Logical True`

2/ Problem when a solver is called two time in the same sif...
   Need to make a copy of the object file to avoid mixing of the saved
   variables in the solver from two different calls:

   `cp $ELMER_HOME/share/elmersolver/lib/FreeSurfaceSolver.so
   MyFreeSurfaceSolver`

   Use a different call in the sif file for Zb:

   `Procedure = "./MyFreeSurfaceSolver" "FreeSurfaceSolver"`
Modelling Tête Rousse Glacier

✓ **Step 1** - Tête Rousse Glacier flow without a water filled-cavity (diagnostic)

✓ **Step 2**
- 2a Influence of an empty cavity below Tête Rousse Glacier (diagnostic)
- 2b Apply a water pressure in the cavity

✓ **Step 3**
- 3a Rate of closure of the cavity for a given drainage scenario (prognostic)
- 3b Add a drainage scenario
Step 3b – Add a drainage scenario

Add an evolution of the water load of the linear form:

\[ h_w = 3170.0 - t \times \Delta h_w / \Delta t \]

**Work to do:**
- write a MATC function hw to prescribe the water load evolution
- write a User Function to do the same (see USF_TR.f90)
Step 3b – Add a drainage scenario

MATC function $hw$ to prescribe the water load evolution:
! Water load function of time (in year)
! Decrease by DH in DT and $h > 3100.0$

```matlab
$ function hw(t) \$
    DH = 70.0;\$
    DT = 20.0;\$
    h = 3170.0 - t*365.25*DH/DT;\$
    if (h > 3100.0) \$
        _hw = h ;\$
    } else \$
        _hw = 3100.0 ;\$
    }\$

Call in the bedrock BC
External Pressure = Variable time, Coordinate 3
Real MATC "rhow*gravity*(hw(tx(0))-tx(1))"
More Steps ???

Some ideas:

- go parallel

- add the Savedata solver to get upper and/or lower surfaces ASCII output

- add the StructuredProjectToPlane solver

Solver 2
Equation = "HeightDepth"
Procedure = "StructuredProjectToPlane" "StructuredProjectToPlane"
Active Coordinate = Integer 3

Operator 1 = depth
Operator 2 = height
End

- write a f90 user function to give the water pressure as a function of time (Step3b)
  For the 2 last points, see Step3c
References