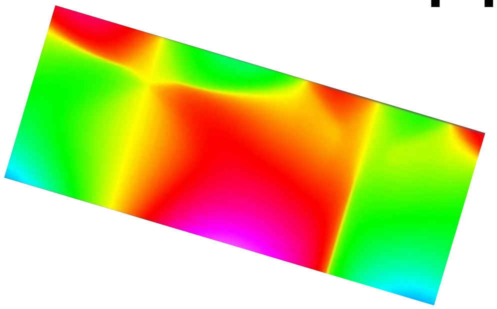
First Elmer/Ice course

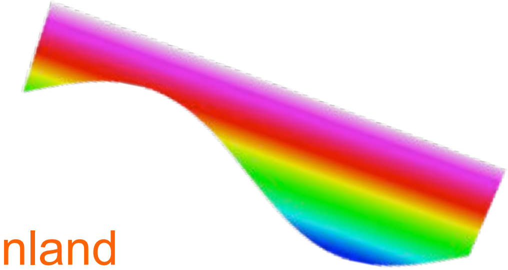
14-15 February 2008 – Updated 2013

Thomas ZWINGER (1) and Olivier GAGLIARDINI (2)

Application to ISMIP HOM(3) tests B and D

Step by step !





(1) CSC-Scientific Computing Ltd., Espoo - Finland
(2) LGGE - Grenoble - France

(3) http://homepages.ulb.ac.be/~fpattyn/ismip/

Outline

Step 0 Start from a very simple test case. What are we solving? (Glen's law, …)

Step 1 Move to test ISMIP-HOM B020 (mesh, periodic BC)

Step 2 Add SaveData solver to get output on the BC (SaveLine)
Add SaveData solver to get cpu and volume of the domain (SaveScalars)
Add ComputeDevStress solver to get the stress field
Add ResultOutput solver to export in vtu format

Step 3 Move to test ISMIP-HOM D020 (sliding law from user function or MATC)

Step 4 Restart from Step 2: Move to Prognostic ISMIP B020.
- Move from a steady to a transient simulation
- Free surface solver

Step 5 Move to Prognostic ISMIP D020.

Step 0

Create a `My_ISMIP_Appli` directory

Copy the directory *Step0 in* *`My_ISMIP_Appli`*

- Make the mesh : `> ElmerGrid 1 2 square.grd`

- Run the test : `> ElmerSolver ismip_step0.sif`

- Watch the results : `> ElmerPost` and open `square\ismip_step0.ep`

- What are we solving?

Stokes:

$$\mathrm{div}\,\boldsymbol{\sigma} + \rho \boldsymbol{g} = 0$$

$$u_{i,i} = 0$$

⬅

Navier-Stokes with convection and acceleration terms neglected :

```
Flow Model = String Stokes
```

in the Stokes solver section

Step 0 – Glen's law and Elmer

In glaciology, you can find (at least) two definitions for Glen's law:

$$D_{ij} = \frac{B}{2}\tau_e^{n-1} S_{ij} \quad ; \quad S_{ij} = 2B^{-1/n}\dot{\gamma}^{(1-n)/n} D_{ij}$$

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

ISMIP notation

where $I_{D_2}^2 = D_{ij}D_{ij}/2$ and $\dot{\gamma}^2 = 2D_{ij}D_{ij}$

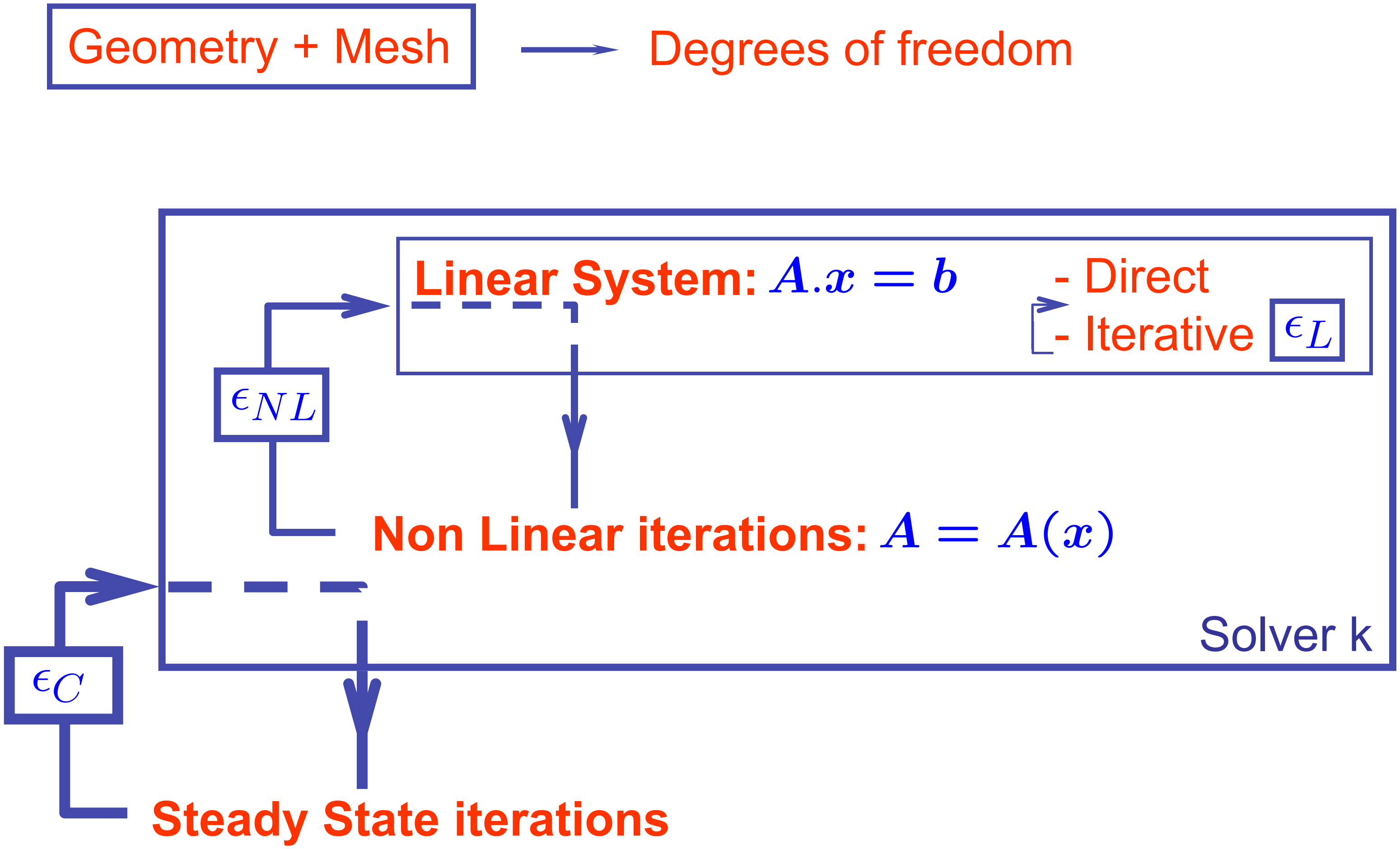
The power-law implemented in Elmer writes: $S_{ij} = 2\eta_0\dot{\gamma}^{m-1} D_{ij}$

$$\boxed{\begin{aligned} &\eta_0 = B^{-1/n} = (2A)^{-1/n} \\ &m = 1/n \\ &\dot{\gamma}^2 \geq \dot{\gamma}_c^2 \end{aligned}}$$

In Material Section:

`Viscosity Model = String "power law"`
`Viscosity = Real` η_0
`Viscosity Exponent = Real` m
`Critical Shear Rate = Real` $\dot{\gamma}_c$

Step 0 – Sketch of a Steady simulation



Geometry + Mesh → Degrees of freedom

Linear System: $\boldsymbol{A.x = b}$

- Direct
- Iterative ϵ_L

ϵ_{NL}

Non Linear iterations: $\boldsymbol{A = A(x)}$

Solver k

ϵ_C

Steady State iterations

$\epsilon_L < \epsilon_{NL} < \epsilon_C$

Step 0 – Numerical methods

In the NS Solver Section: (see Chapters 3 and 4 of Elmer Solver Manual)

- Solution for the Linear System:

```
Linear System Solver = Direct
Linear System Direct Method = umfpack
```

- Non-Linear System :

Picard

Newton

```
Nonlinear System Max Iterations = 100
Nonlinear System Convergence Tolerance  = 1.0e-5
Nonlinear System Newton After Iterations = 5
Nonlinear System Newton After Tolerance = 1.0e-02
Nonlinear System Relaxation Factor = 1.00
```

$= \epsilon_{NL}$

- Coupled problem (not needed here in fact…):

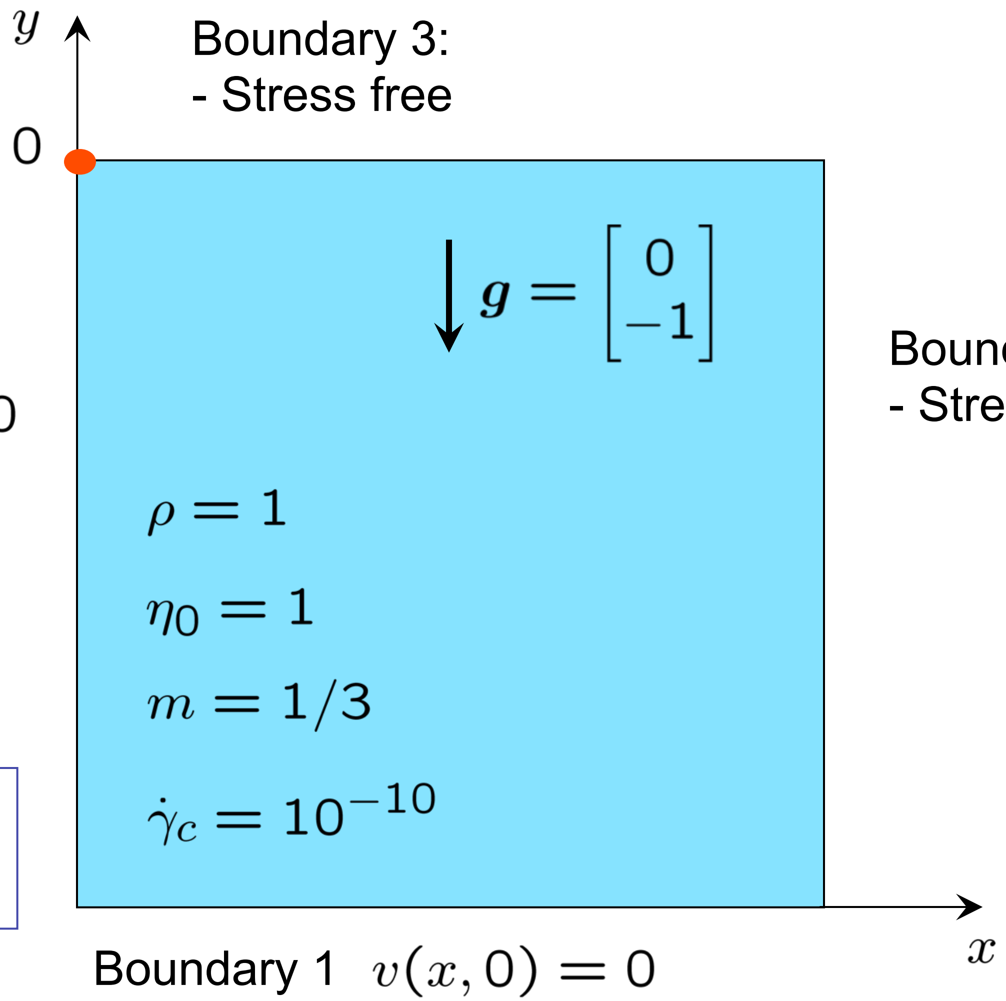
```
Steady State Convergence Tolerance = Real 1.0e-3
```

$= \epsilon_{C}$

- Stabilization of the Stokes equations:

`Stabilization Method = String Bubbles` (other options: `Stabilized, P2P1`)

Step 0 – What are we solving?



Boundary 3:
- Stress free

$p(0,1) = 0$

$\boldsymbol{g} = \begin{bmatrix} 0 \\ -1 \end{bmatrix}$

Boundary 4
$u(0,y) = 0$

Boundary 2:
- Stress free

$\rho = 1$

$\eta_0 = 1$

$m = 1/3$

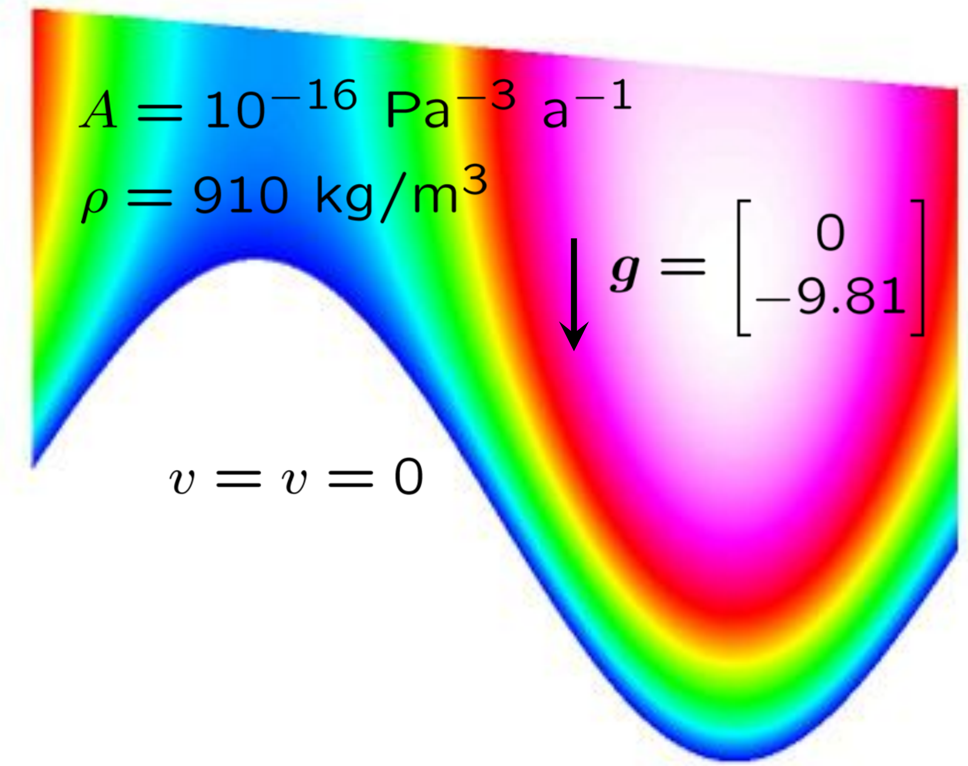
$\dot{\gamma}_c = 10^{-10}$

Mesh:
20 x 20 Q4 elements

Boundary 1 $v(x,0) = 0$

Step 1 – Move to ISMIP-HOM B020

What we have to solve :



Stress free $z_s(x) = -x\,\mathrm{tan}(0.5°)$

Periodic BC

$A = 10^{-16}\ \mathrm{Pa}^{-3}\ \mathrm{a}^{-1}$

$\rho = 910\ \mathrm{kg/m}^3$

$\boldsymbol{g} = \begin{bmatrix} 0 \\ -9.81 \end{bmatrix}$

Periodic BC

$v = v = 0$

$L = 20 \times 10^3\ \mathrm{m}$

$$z_b(x) = z_s(x) - 1000 + 500\,\mathrm{sin}(\frac{2\pi}{L}x)$$

Step 1 – Changes from Step0

- New directory, new names !

  e.g.: `square.grd -> rectangle.grd`

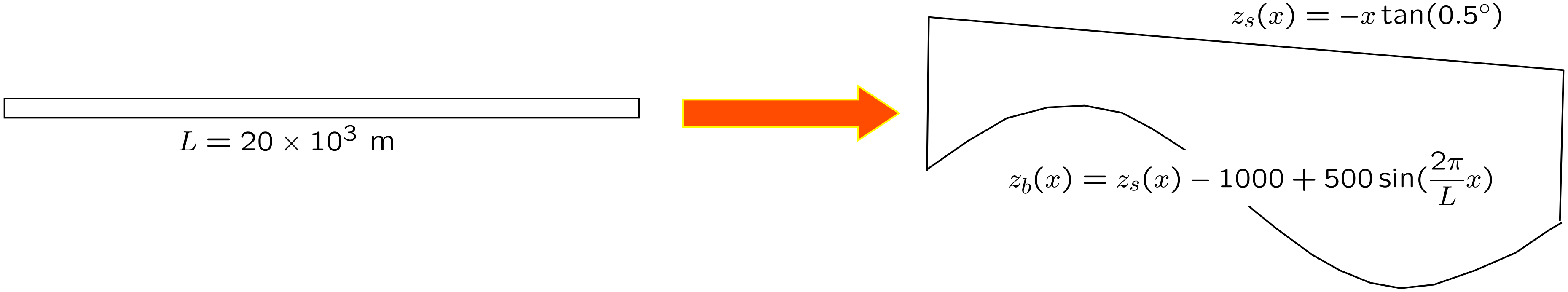
- Make the mesh : use of the `StructuredMeshMapper` solver

- Right values for the different constants (which system of Units ?)

- Add the periodic boundary conditions

Step 1 – StructuredMeshMapper

- Start from rectangular mesh L x 1m and use the solver StructuredMeshMapper to produce the ISMIP B geometry



- Add the solver StructuredMeshMapper

```
Solver 1
  Equation = "MapCoordinate"
  Procedure = "StructuredMeshMapper" "StructuredMeshMapper"
  Active Coordinate = Integer 2
  Mesh Velocity Variable = String ”dSdt”
  Mesh Update Variable = String ”dS”
  Mesh Velocity First Zero = Logical True
End
```

(Not really needed in steady)

Step 1 – StructuredMeshMapper

Specify in the Boundary conditions 1 and 3 what is the mapping for the Bottom and Top surfaces, respectively:

```
$Slope = 0.5 * pi / 180.0
$L = 20000.0

! Bedrock
Boundary Condition 1
  Target Boundaries = 1
  Velocity 1 = Real 0.0e0
  Velocity 2 = Real 0.0e0
  Bottom Surface = Variable Coordinate 1
    Real MATC "-tx*tan(Slope)-1000.0+500.0*sin(2.0*pi*tx/L)"
End

! Upper Surface
Boundary Condition 3
  Target Boundaries = 3
  Top Surface = Variable Coordinate 1
    Real MATC "-tx*tan(Slope)"
End
```

Step 1 – Elmer and Units

The choice of Units have to be coherent.
But you are free because the Stiff matrix is normalized.

For the Stokes problem, one should give values for:

- the density: ρ $(= 910\ \mathrm{kg/m^3})$
- the gravity: g $(= 9.81\ \mathrm{m\ s^{-2}})$
- the viscosity: η_0 $(\mathrm{Pa\ s^{1/n}})$ $(1\ \mathrm{Pa} = 1\ \mathrm{kg\ s^{-2}\ m^{-1}})$

kg – m – s [SI] : velocity in m/s and timestep in secondes ☹

kg – m – a : velocity in m/a and timesteps in years ☺ $1\ \mathrm{a} = 31\,557\,600\ \mathrm{s}$

MPa – m – a : velocity in m/a and Stress in MPa ☺

(What I will use in the following)

Step 1 – Value of the ISMIP constants

For ISMIP tests A-D, the value for the constants are

- the density: $\rho = 910 \text{ kg/m}^3$
- the gravity: $g = 9.81 \text{ m s}^{-2}$
- the fluidity: $A = 10^{-16} \text{ Pa}^{-3} \text{ a}^{-1}$

	USI kg - m - s		kg - m - a		MPa - m - a	
g =	9.81	m / s^2	9.7692E+15	m / a^2	9.7692E+15	m / a^2
ρ =	910	kg / m^3	910	kg / m^3	9.1380E-19	MPa m^{-2} a^2
A =	3.1689E-24	kg^{-3} m^3 s^5	1.0126E-61	kg^{-3} m^3 a^5	100	MPa^{-3} a^{-1}
η =	5.4037E+07	kg m^{-1} s$^{-5/3}$	1.7029E+20	kg m^{-1} a$^{-5/3}$	0.1710	MPa a$^{1/3}$

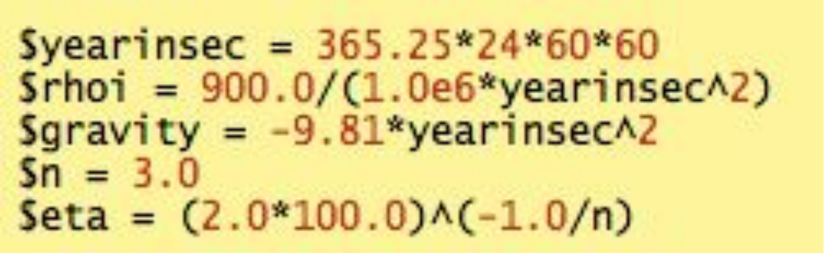
$$\eta_0 = B^{-1/n} = (2A)^{-1/n}$$

$$m = 1/n$$

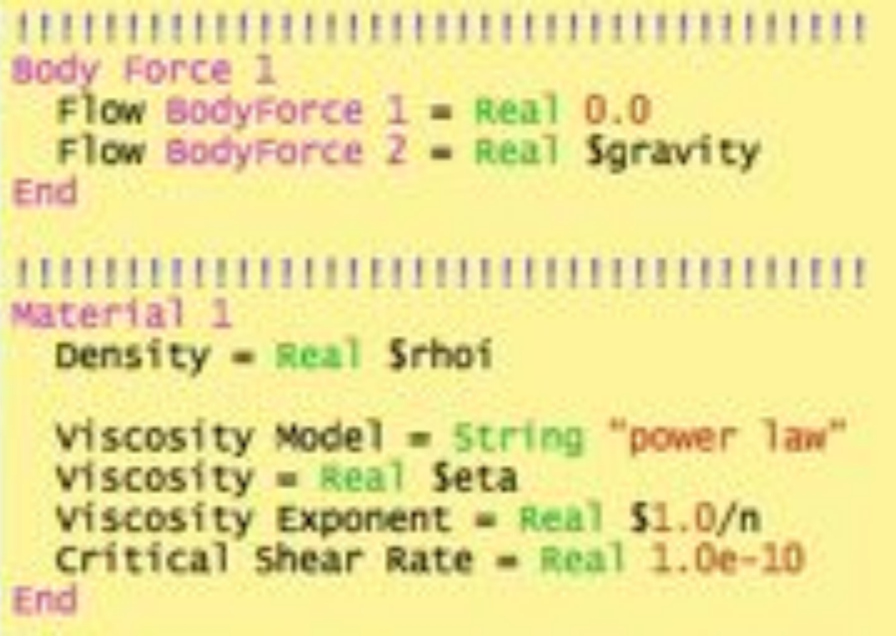
$$\dot{\gamma}^2 \geq \dot{\gamma}_c^2$$

Step 1 – Value of the ISMIP constants

One can use MATC coding to get the correct value of the parameters

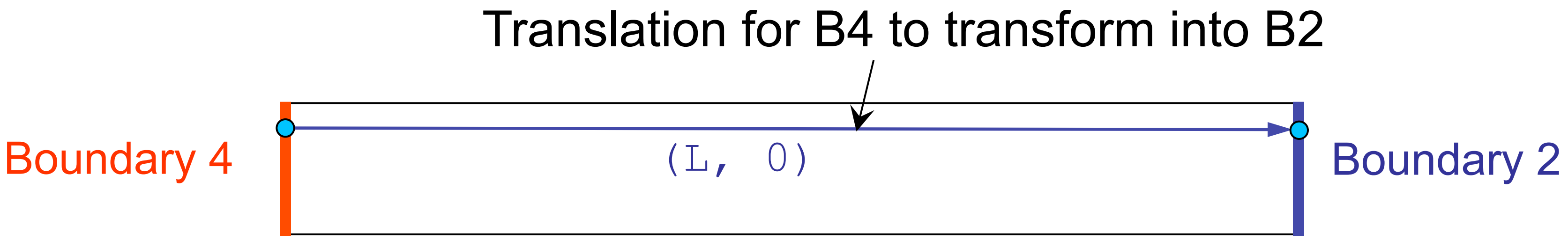


```
$yearinsec = 365.25*24*60*60
$rhoi = 900.0/(1.0e6*yearinsec^2)
$gravity = -9.81*yearinsec^2
$n = 3.0
$eta = (2.0*100.0)^(-1.0/n)
```



```
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
Body Force 1
  Flow BodyForce 1 = Real 0.0
  Flow BodyForce 2 = Real $gravity
End

!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
Material 1
  Density = Real $rhoi

  Viscosity Model = String "power law"
  Viscosity = Real $eta
  Viscosity Exponent = Real $1.0/n
  Critical Shear Rate = Real 1.0e-10
End
```

Step 1 – Periodic boundary conditions



Declare at the top of the sif:

```
$L = 20.0e3
        …
Boundary Condition 2
  Target Boundaries = 2
  Periodic BC = 4
  Periodic BC Translate(2) = Real $L 0.0
  Periodic BC Velocity 1  = Logical True
  Periodic BC Velocity 2  = Logical True
  Periodic BC Pressure  = Logical True
End
Boundary Condition 4
  Target Boundaries = 4
End
```

Nothing to declare for BC4 !

Step 2 – Add ComputeDevStress

Objective: compute the stress field as $\int_V S_{ij}\Phi\,\mathrm{d}V = 2\int_V \eta D_{ij}\Phi\,\mathrm{d}V$

where D_{ij} and η are calculated from the nodal velocities using the derivative of the basis functions

- Add a Solver

```
Solver 3
  Equation = Sij
  Variable = -nooutput "Sij"
  Variable DOFs = 1
  Exported Variable 1 = Stress[Sxx:1 Syy:1 Szz:1 Sxy:1]
  Exported Variable 1 DOFs = 4
  Stress Variable Name = String "Stress"
  Procedure = "ElmerIceSolvers" "ComputeDevStress"
  Flow Solver Name = String "Flow Solution"
  Linear System Solver = Direct
  Linear System Direct Method = umfpack
End
```

- Add in the material section:

```
Cauchy = Logical False
```

Step 2 – Add SaveData Solver (SaveLine)

Objective: save the variables on the top surface (ASCII matrix file)

- Add a new solver

```
Solver 4
  Exec Solver =  After All
  Procedure = File “SaveData” “SaveLine”
  Filename =  “ismip_surface.dat”
  File Append = Logical False
End
```

- Tell in which BC you want to save the data

```
Boundary Condition 3
  Target Boundaries = 3
  Save Line = Logical True
End
```

- Ordering of the variables: see file `ismip_surface.dat.names`

Step 2 – Add SaveData Solver (SaveLine)

SaveLine can also be used to save data at a ‘drilling site’ (a line which is not a boundary). Here, the data are saved at x = 10km.

- Change the solver section

```
Solver 4
  Exec Solver =  After All
  Procedure = File “SaveData” “SaveLine”
  Filename =  “ismip_drilling.dat”
  Polyline Coordinates(2,2) = Real $ (0.5*L) -1000. (0.5*L) 0.0
  File Append = Logical False
End
```

- And don’t forget to comment the `Save line = Logical True` in BC3

Step 2 – Add SaveScalars

SaveScalars allows to save scalars and derived quantities. Here, we will save:

1/ the volume of the domain (surface),
2/ the maximum value of the absolute horizontal velocity,
3/ the flux on the 3 boundaries 2, 3 and 4.
4/ the CPU time,
5/ the CPU memory

Step 2 – Add SaveScalars

-Add a new solver

```
Solver 5
  Exec Solver =  After
  Procedure = “SaveData” “SaveScalars”
  Filename = “ismip_scalars.dat”
  File Append = Logical True
  Variable 1 = String “flow solution”
  Operator 1 = String “Volume”
  Variable 2 = String “Velocity 1”
  Operator 2 = String “max abs"
  Variable 3 = String “flow solution”
  Operator 3 = String “Convective flux”
  Operator 4 = String “cpu time”
  Operator 5 = String “cpu memory”
End
```

- Tell at which boundaries you want to save the flux

```
Flux Integrate = Logical True
```

Step 3 – Add ResultOutput

- ResultOutput allows to export the result in vtu format and use Paraview for post-treatment

```
Solver 6
  Exec Solver = After TimeStep
  Exec Interval = 1
  Equation = "result output"
  Procedure = "ResultOutputSolve" "ResultOutputSolver"
  Output File Name = String "ismip$Step".vtu"
  Output Format = String vtu
End
```

- For all these added solvers, modify the Equation section:

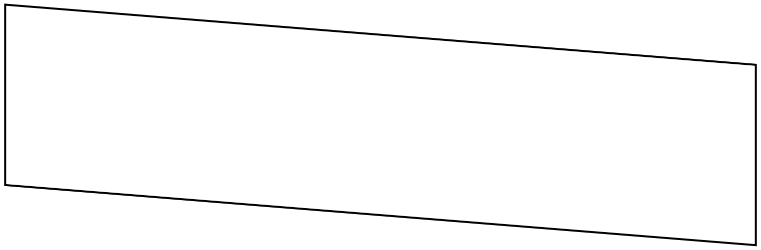
```
Equation 1
  Active Solvers(6) = 1 2 3 4 5 6
End
```

Step 3 – Move to ISMIP-HOM D020

Changes from B020:

- geometry of domain

$$\begin{cases} z_s(x,y) = -x\,\mathrm{tan}(0.1^\circ) \\ z_b(x,y) = z_s(x,y) - 1000 \end{cases}$$



→ modify the `Top Surface` and `Bottom Surface` variables

- boundary condition at the bedrock interface

$$\begin{cases} \tau_{nt} = \beta^2 u_t \\ u_n = \boldsymbol{u} \cdot \boldsymbol{n} = 0 \end{cases}$$

with $\beta^2(x) = 1000 + 1000\,\mathrm{sin}(\frac{2\pi}{L}x)$

in [Pa a m^{-1}] !

→ modify the Boundary Condition 1

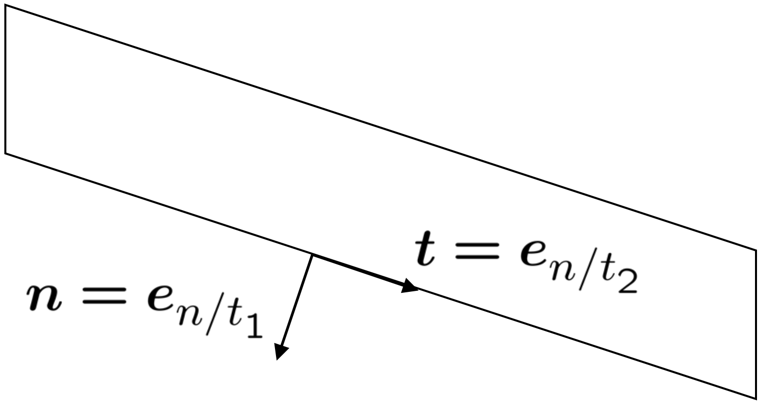
Step 3 – Move to ISMIP-HOM D020

Friction law in Elmer:

$$C_i u_i = \sigma_{ij} n_j \ (i = 1, 2)$$

$$\Rightarrow \quad C_t u_t = \sigma_{nt} \ ; \ C_n u_n = \sigma_{nn}$$

where $\boldsymbol{n}$ is the surface normal vector



Modification of the Boundary Condition 1:

- First Solution: MATC definition of Ct

```
Boundary Condition 1
  Target Boundaries = 1
  Flow Force BC = Logical True
  Normal-Tangential Velocity = Logical True

  Velocity 1 = Real 0.0e0

  Slip Coefficient 2 = Variable coordinate 1
    Real  MATC "1.0e-3*(1.0 + sin(2.0*pi* tx / L ))
End
```

} Stress condition defined in a normal-tangential coordinate system

} $u_n = 0$

} $C_t = \ldots$

in [MPa a m^{-1}] !

Step 3 – Move to ISMIP-HOM D020

- Second Solution: User Function to define Ct

```
Boundary Condition 1
  …
  Slip Coefficient 2 = Variable coordinate 1
     Real Procedure "./ISMIP_D" "Sliding"
End
```

where Sliding is a User Function defined in the file `ISMIP_D.f90` (see next slide)

Compilation:

```
> elmerf90 ISMIP_D.f90 -o ISMIP_D
```

Step 3 – Move to ISMIP-HOM D020

```
FUNCTION Sliding ( Model, nodenumber, x) RESULT(C)
   USE Types

   IMPLICIT NONE
   TYPE(Model_t) :: Model
   INTEGER :: nodenumber, i
   REAL(KIND=dp) :: x, C, L
   LOGICAL :: FirstTime=.True.

   SAVE FirstTime, L

   IF (FirstTime) THEN
       FirstTime=.False.
       L = MAXVAL(Model % Nodes % x)
   END IF

   x = Model % Nodes % x(nodenumber)
   C  =  1000.0e-6_dp*(1.0_dp + SIN(2.0_dp * Pi * x/ L))
                       ! in MPa a /m
END FUNCTION Sliding
```

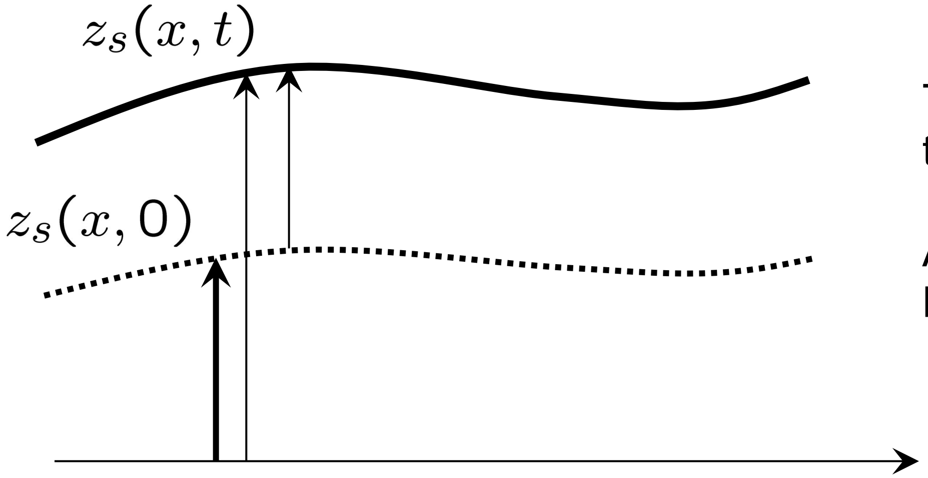
Step 4 – Move to prognostic B020

Move from a Diagnostic to a prognostic simulations:

- Steady to transient

- Add the **free surface** solver

$$\frac{\partial z_s}{\partial t} + u_x \frac{\partial z_s}{\partial x} - u_z = a$$



The mesh is vertically deformed using the StructuredMeshMapper solver.

An alternative is to use the MeshUpdate solver (see older courses)

Step 4 – Steady to transient

The simulation Section has to be modified:

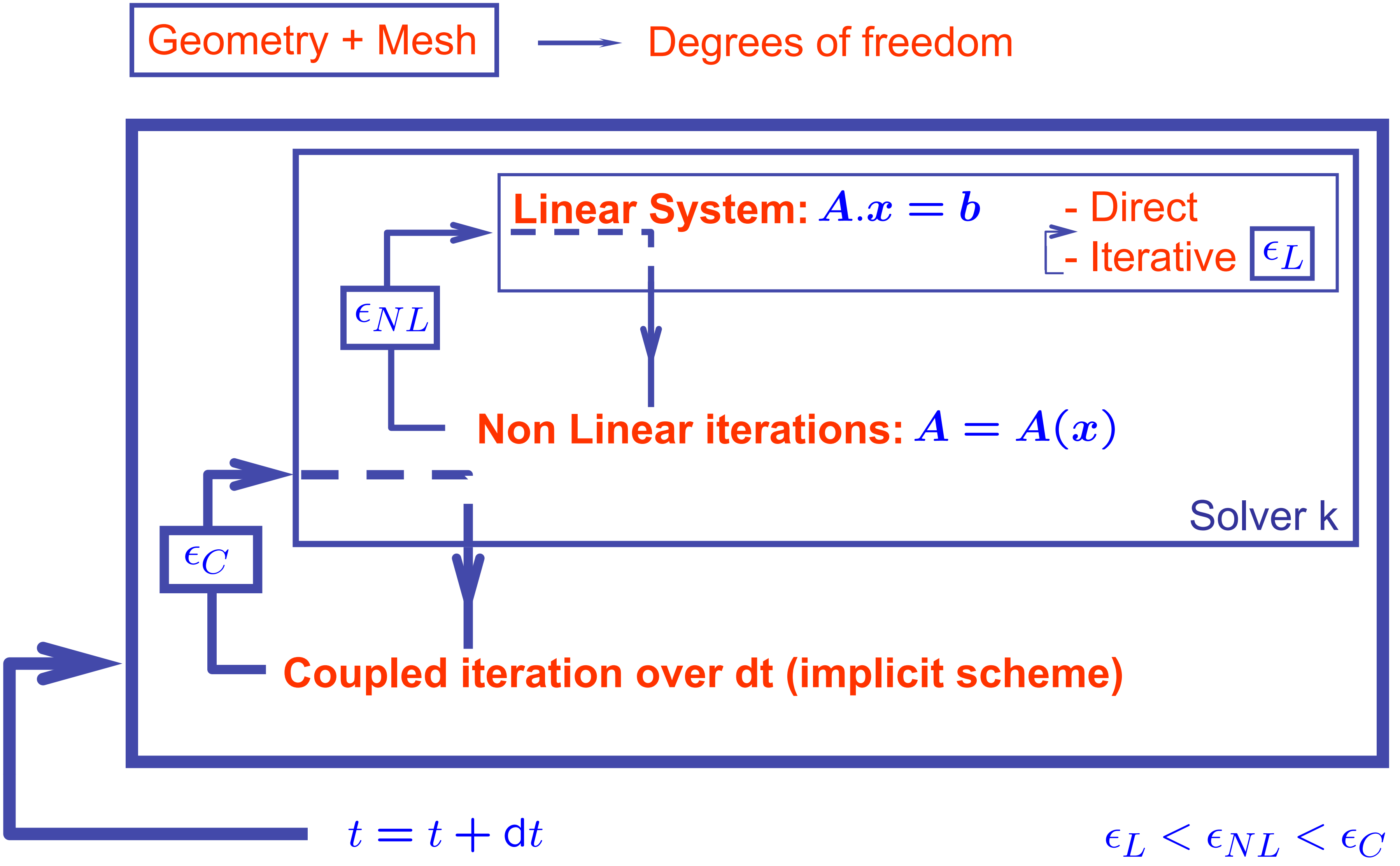
```
Simulation Type = Transient

Timestepping Method = “bdf”
BDF Order = 1
Output Intervals = 1
Timestep Intervals = 200
Timestep Sizes = 1.0

Steady State Min Iterations = 1
Steady State Max Iterations = 10
```

Backward Differences Formulae

Save in .ep file

To control the “implicity” of the solution over one time step
(see example bellow).

Step 4 – Sketch of a transient simulation



Geometry + Mesh → Degrees of freedom

Linear System: $\boldsymbol{A.x = b}$

- Direct
- Iterative $\boxed{\epsilon_L}$

$\boxed{\epsilon_{NL}}$

Non Linear iterations: $\boldsymbol{A = A(x)}$

Solver k

$\boxed{\epsilon_C}$

Coupled iteration over dt (implicit scheme)

$t = t + \mathrm{d}t$

$\epsilon_L < \epsilon_{NL} < \epsilon_C$

Step 4 – Free surface Solver

The free surface solver only apply to the boundary 3 (top 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.

Tell in BC3 that this is the body 2:

```
Boundary Condition 3
  Target Boundaries = 3
        …
  !!! this BC is equal to body no. 2 !!!
  Body Id = 2
        …
End
```

Step 4 – Free surface Solver

Add the Free Surface Solver:

The minimum is presented here, you can add limits not to be penetrated by the free surface

```
Solver 2
   Equation = "Free Surface"
   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-5
   Linear System Abort Not Converged = False
   Linear System Residual Output = 1

   Steady State Convergence Tolerance = 1.0e-03

   Relaxation factor = Real 1.0
   Stabilization Method = Bubbles
   End
          …
```

Step 4 – Free surface Solver

Body Force 2:

```
Body Force 2
  Zs Accumulation Flux 1 = Real 0.0e0
  Zs Accumulation Flux 2 = 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: give $z_s(x,0)$

```
Initial Condition 2
  Zs = Variable Coordinate 1
     Real MATC “-tx*tan(Slope)”
End
```

Step 4 – StructuredMeshMapper

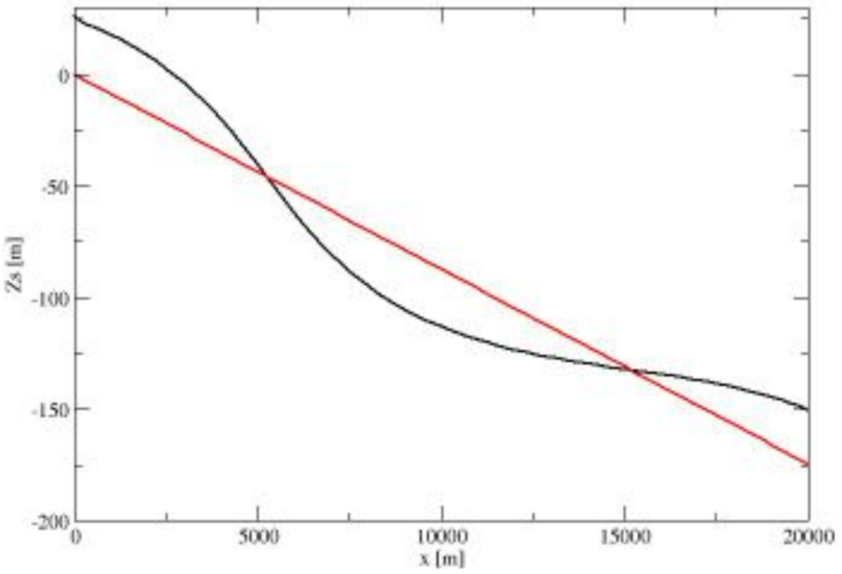
- The `Top Surface` variable is now equals to the variable `Zs`

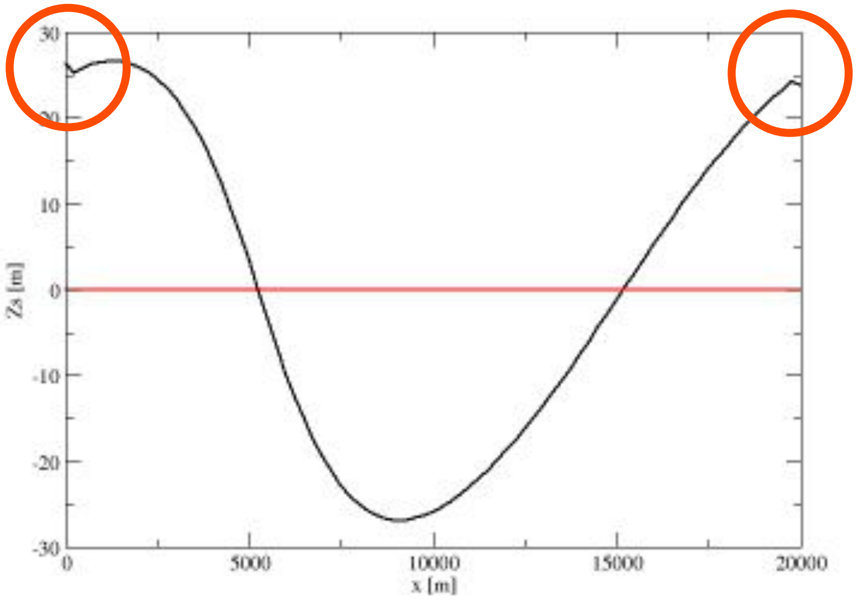
```
Solver 1
  Equation = "MapCoordinate"
  Procedure = "StructuredMeshMapper" "StructuredMeshMapper"
  Active Coordinate = Integer 2

  Mesh Velocity Variable = String  "dSdt"
  Mesh Update Variable = String  "dS »
  Mesh Velocity First Zero = Logical True

  Top Surface Variable = String "Zs"
  Dot Product Tolerance = Real 1.0e-3
End
```

- And delete in the top surface BC the definition of `Top Surface`.

Step 4 – Results !

Comparison of the initial and steady surface of the prognostic run



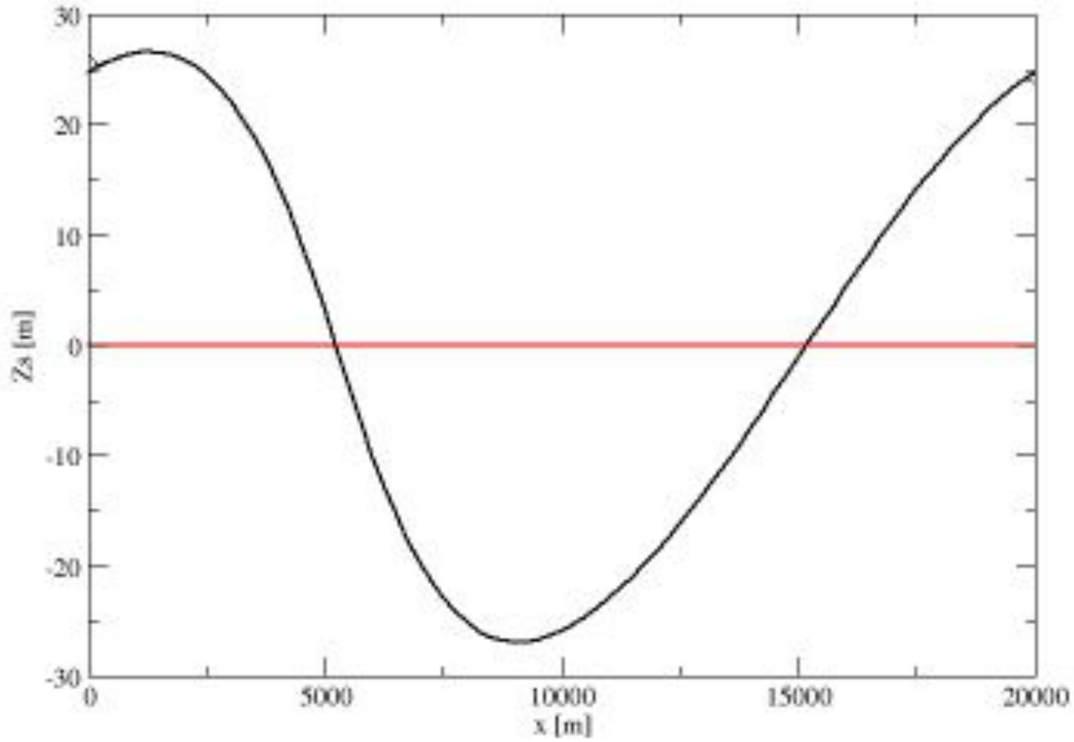


Not periodic

Step 4 – better results !

Turn the mesh so that zs = 0 (turn the gravity vector also !)
Force zs to be periodic

See Step4_hori



Step 5 – Move to prognostic D020

Merge Step 3 and Step 4 and it should work !