Elmer/Ice package

Glaciologická nástavba Elmeru

V tuto chvíli zkompilovaná pod ifort, ifort+mkl, gfortran na geofos, geofy1

http://elmerice.elmerfem.org/wiki/doku.php?id=start

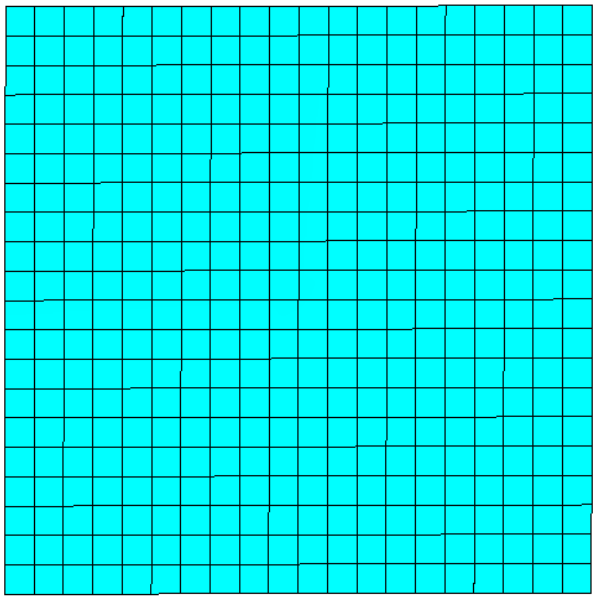
Příklad (pro připomenutí)

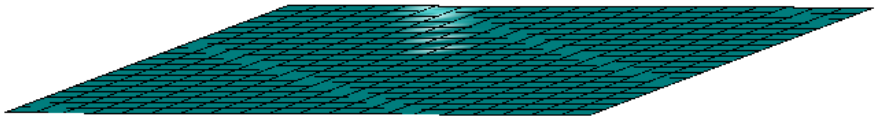
Benchmark experiment z ISMIP-HOM (
http://homepages.ulb.ac.be/~fpattyn/ismip/ismiphom.pdf)
Experiment C-160
Zdroj:
http://elmerice.elmerfem.org/wiki/lib/exe/fetch.php?media=solvers:test_ssa.tar.gz

Obsahuje mesh_C.grd:

```
 ***** ElmerGrid input file for structured grid generation *****
Version = 210903
Coordinate System = Cartesian 3D
Subcell Divisions in 3D = 1 1 1
Subcell Limits 1 = 0.0  1.60000000e+05
Subcell Limits 2 = 0.0  1.60000000e+05
Subcell Limits 3 = 0.0 1.0
Material Structure in 2D
  1
End
Materials Interval = 1 1
Boundary Definitions
! type     out      int
  1       -1        1        1
  2       -2        1        1
  3       -3        1        1
  4       -4        1        1
End
!Advanced Elements
!!mat n e f d b
!  1  1 0 0 0 8
!End
Numbering = Horizontal
Element Degree = 1
Element Innernodes = False
Coordinate Ratios = 1
Decimals = 12
Element Divisions 1 = 20
Element Divisions 2 = 20
Element Divisions 3 = 10
```

Překladem: **ElmerGrid 1 2 mesh_C.grd**
vytvoříme síť





Podle pokynů v README síť zdeformujeme pomoci pomocného programu, který volá explicitní funkce popisující horni a spodní rozhraní

elmerf90-nosh MshGlacierSynthetic.f90 fbed.f90 fsurf.f90 -o MshGlacierSynthetic
./MshGlacierSynthetic

fbed.f90:

```
FUNCTION fbed(x,y).
USE types
IMPLICIT NONE
REAL(KIND=dp) :: x, y, fbed
REAL(KIND=dp) :: fsurf
fbed = fsurf(x,y) -1000.0d0
END FUNCTION fbed
```

fsurf.f90

```
FUNCTION fsurf(x,y).
USE types
IMPLICIT NONE
REAL(KIND=dp) :: x, y, fsurf
fsurf = - x * TAN(0.1d0*Pi/180.0d0)..
END FUNCTION fsurf
```

Řešení v SSA přiblížení:

- 1) Run the `GetMeanValueSolver` routine to infer the vertically integrated value of the viscosity and the mean density.
- (2) Compute the SSA solution on the bedrock boundary using the `SSABasalSolver` solver.
- (3) Export vertically the solution previously computed on the bedrock over the whole domain. This can be either done using the `SSASolver` routine (no vertical variation of the velocities) or the "SIA Solver" using the `SSABasalFlow` solution as a Dirichlet boundary condition for the SIA velocity.

Vlastní model je obsažen v ssa.sif

```
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!                                           !!
!! Test SSA Solver                           !!
!!                                           !!
!! Test ISMIP C160                           !!
!!                                           !!
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!

check keywords warn
echo on

$L = 160.0e3
$dz = -L * tan(0.1*pi/180.0)

$yearinsec = 31556926.0
$rhoi = 910.0/(1.0e6*yearinsec^2)
$gravity = -9.81*yearinsec^2
$A = 100.0 ! MPa-3a-1
$n = 3.0
$etai = 1.0/(2.0*A)^(1.0/n)

Header
  Mesh DB "." "mesh_C"
End

Constants
End

!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
Simulation
  Coordinate System  = Cartesian 3D
  Simulation Type = Steady State

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

  Post File = "ismip_C_SSA.ep"
  max output level = 100
End

!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
Body 1
  Equation = 1
  Body Force = 1
  Material = 1
  Initial Condition = 1
End
```

```
! The Bedrock
Body 2
  Equation = 2
  Body Force = 1
  Material = 1
  Initial Condition = 1
End

!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
Initial Condition 1
  Depth = Real 0.0
  SSABasalFlow 1 = Real 100.0
  SSABasalFlow 2= Real 0.0
End

!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
Body Force 1
  Flow BodyForce 1 = Real 0.0
  Flow BodyForce 2 = Real 0.0
  Flow BodyForce 3 = Real $gravity
End

!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
Material 1
  Viscosity Model = String "power law"
  Density = Real $rhoi
  Viscosity = Real $etai
  Viscosity Exponent = Real $1.0/n
  Critical Shear Rate = Real 1.0e-10
End

!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
Solver 1
  Equation = "Flowdepth"
   Procedure = File "ElmerIceSolvers" "FlowDepthSolver"
   Variable = String "Depth"
   Variable DOFs = 1
   Linear System Solver = "Direct"
   Linear System Direct Method = umfpack
   ! this sets the direction
   ! -1 is negative z-direction (upside down)
   ! +1 is positive (downside up)
   Gradient = Real -1.0E00
  ! switch that to True, if you
  ! want to have free surface gradients
  ! to be computed
  !-------------------------------------
  Calc Free Surface = Logical True
  ! will contain the variable of the corresponding free surface position
  !--------------------------------------------------------------------
  Exported Variable 1 = String "FreeSurf"
  Exported Variable 1 DOFs = 1
  FreeSurf Name = String "FreeSurf"
  ! next two will contain values of the free surface gradient
  ! nice to be used in post-processing for evaluation of the
  ! surface mass balance in diagnostic runs
  !-----------------------------------------------------------
  Exported Variable 2 = String "FreeSurfGrad1"
  Exported Variable 2 DOFs = 1
  Exported Variable 3 = String "FreeSurfGrad2"
  Exported Variable 3 DOFs = 1
End

Solver 2
  Equation = "SSA-IntValue"
  Procedure = File "ElmerIceSolvers" "GetMeanValueSolver"
  Variable = -nooutput String "Integrated variable"
```

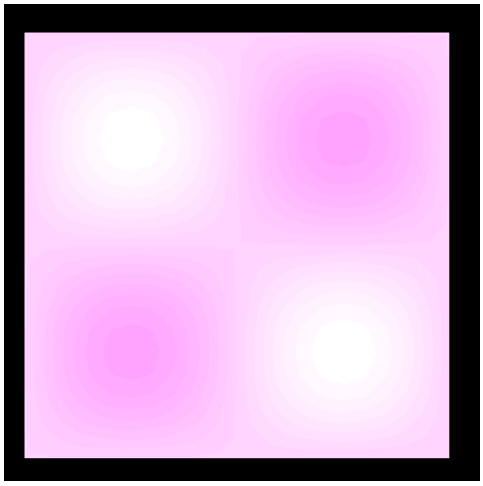
```
  Variable DOFs = 1

  Exported Variable 1 = String "Integrated Viscosity"
  Exported Variable 1 DOFs = 1
  Exported Variable 2 = String "Mean Density"
  Exported Variable 2 DOFs = 1

  Linear System Solver = Direct
  Linear System Direct Method = umfpack

  Steady State Convergence Tolerance = Real 1.0e-3
End
!
Solver 3
  Equation = "SSA-BasalVelo"
  Procedure = File "ElmerIceSolvers" "SSABasalSolver"
  Variable = String "SSABasalFlow"
  Variable DOFs = 2   ! 2 in 3D

  Linear System Solver = Direct
  Linear System Direct Method = umfpack

  Nonlinear System Max Iterations = 100
  Nonlinear System Convergence Tolerance  = 1.0e-5
  Nonlinear System Relaxation Factor = 1.00

  Steady State Convergence Tolerance = Real 1.0e-3
End

Solver 4
  Equation = "SIA Velocity"
  Procedure = File "ElmerIceSolvers" "SIASolver"
  Variable = -nooutput String "varSIA"
  Variable DOFs = 1

  Exported Variable 1 = String "SIAFlow"
  Exported Variable 1 DOFs = 4  ! 3 in 2D, 4 in 3D

  Linear System Solver = Direct
  Linear System Direct Method = umfpack

  Steady State Convergence Tolerance = Real 1.0e-3
End

Equation 1
  Active Solvers(3) = 1 2 4
End

Equation 2
  Active Solvers(1) = 3
End


!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!! BC y=y0
Boundary Condition 1
  Target Boundaries = 1
End

!!! BC x=xmax
Boundary Condition 2
  Target Boundaries = 2
  Periodic BC = 4
  Periodic BC Translate(3) =  $(L)  (0.0)  (dz)
  Periodic BC SSABasalFlow 1  = Logical True
  Periodic BC SSABasalFlow 2  = Logical True
End
```

```
!!! BC y=ymax
Boundary Condition 3
  Target Boundaries = 3
  Periodic BC = 1
  Periodic BC Translate(3) = $(0.0)  (L)  (0.0)
  Periodic BC SSABasalFlow 1  = Logical True
  Periodic BC SSABasalFlow 2  = Logical True
End

!!! BC x=x0
Boundary Condition 4
  Target Boundaries = 4
End

!!! bedrock
Boundary Condition 5
  Target Boundaries = 5
  Body Id = Integer 2

  SIAFlow 1 = Equals SSABasalFlow 1
  SIAFlow 2 = Equals SSABasalFlow 2
  SIAFlow 3 = Real 0.0e0
  SSA Slip Coefficient 1 = Variable Coordinate 1, Coordinate 2
     Real MATC "1000.0e-6*(1.0+sin(2.0*pi*tx(0)/L)*sin(2.0*pi*tx(1)/L))"
  SSA Slip Coefficient 2 = Variable Coordinate 1, Coordinate 2
     Real MATC "1000.0e-6*(1.0+sin(2.0*pi*tx(0)/L)*sin(2.0*pi*tx(1)/L))"
End

!!! Upper free surface
Boundary Condition 6
  Target Boundaries = 6
  Depth = Real 0.0
  Integrated Viscosity = Real 0.0
  Mean Density = real 0.0
  SIAFlow 4 = Real 0.0     ! p=0 at the bottom
End

!!! Point (0, L/4, -1000.0) v = 0
Boundary Condition 7
  Target Coordinates(1,3) = Real $(0.0) (0.25*L) (0.0)
  SSABasalFlow 2 = Real 0.0
End
```

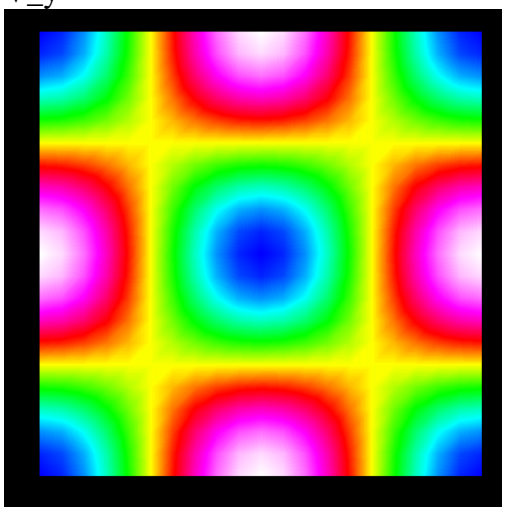
Spustíme pomocí

ElmerSolver ssa.sif

Výsledek (vizualizace ElmerPost):
V_x



V_y



V_z