Elmer/Ice advanced course

Simple Hydrological Model

Contents

The aim of this exercise is to compute a simplified hydrological potential from given geometry

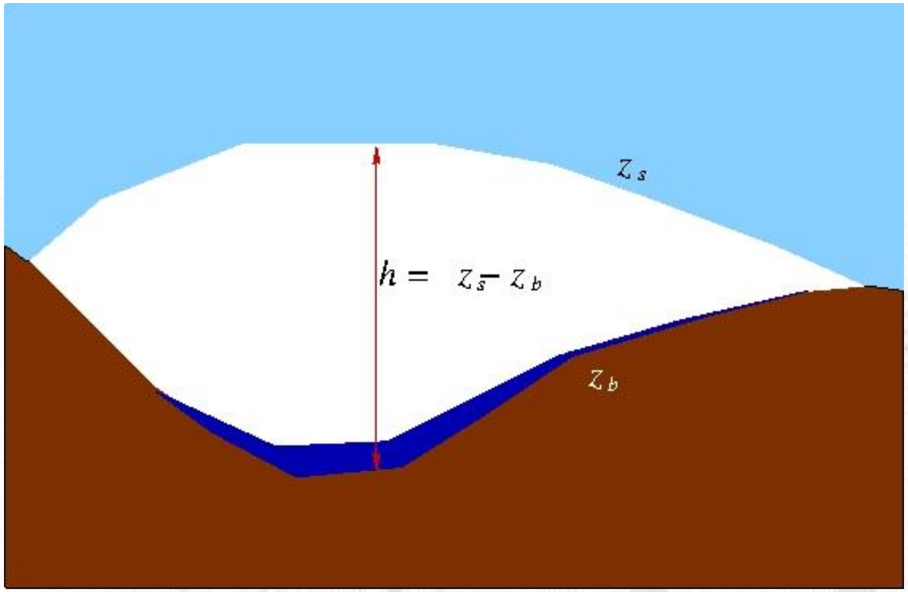
- Definition of hydrological potential
- Steps of implementation in a user defined solver
  - Principle structure
  - Connection to Solver Input File
  - Compilation
- Setup of Solver Input File

Potential

- Free surface: z_s
- Bedrock: z_b
- Thickness: $h = z_s - z_b$
- Hydraulic Potential:

$$\Phi = \varrho_w g z_b + p_w$$

- Water pressure: $p_w = \varrho_i g h - N$
- Effective pressure: $N \approx 0 \Rightarrow p_w \approx \varrho_i g h$

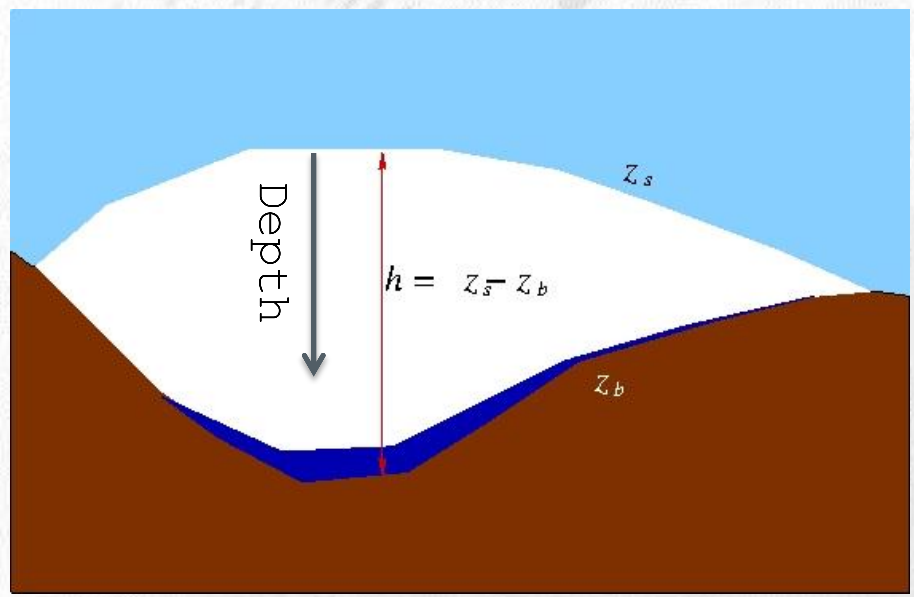


$$\boxed{\Phi = [\varrho_i z_s + (\varrho_w - \varrho_i) z_b] g}$$

Potential

- In Elmer: `Depth`
- @ bedrock: `Depth`= h
- Write routine that solves for:
  - Reads existing variable Depth
  - Reads in Material parameters $\varrho_w, \ \varrho_i$
  - Reads in gravity from body force g

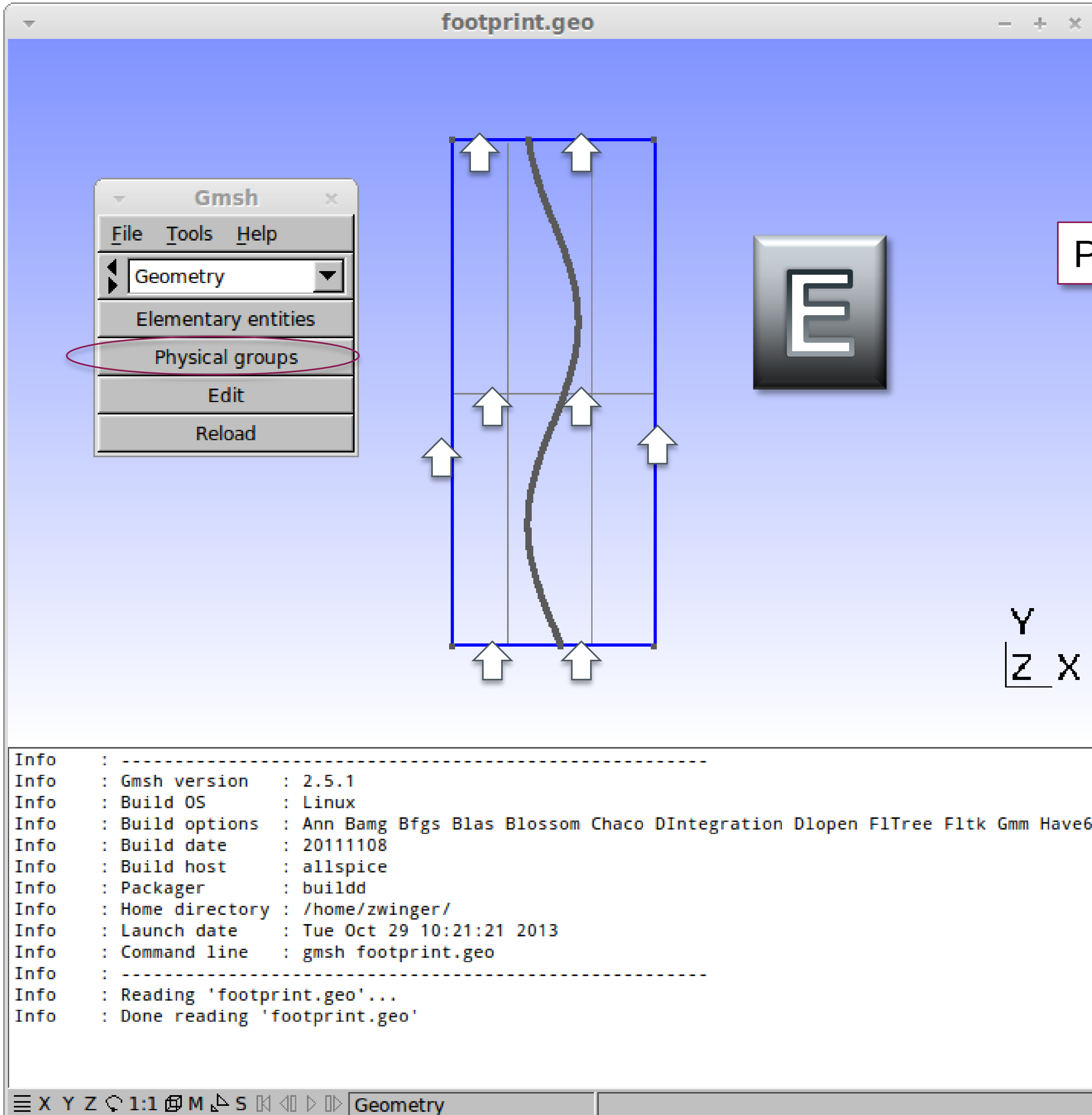
$$\Phi = [\varrho_i h + \varrho_w z_b] g$$



Prerequisites

- Instructions to do everything from scratch found in the README file
- The material contains a Python script that would produce a DEM and a footprint mesh
- We start from an existing DEM and footprint mesh in GMSH format
- Open the mesh:> **`cd Mesh`**

  > **`gmsh footprint.geo`**



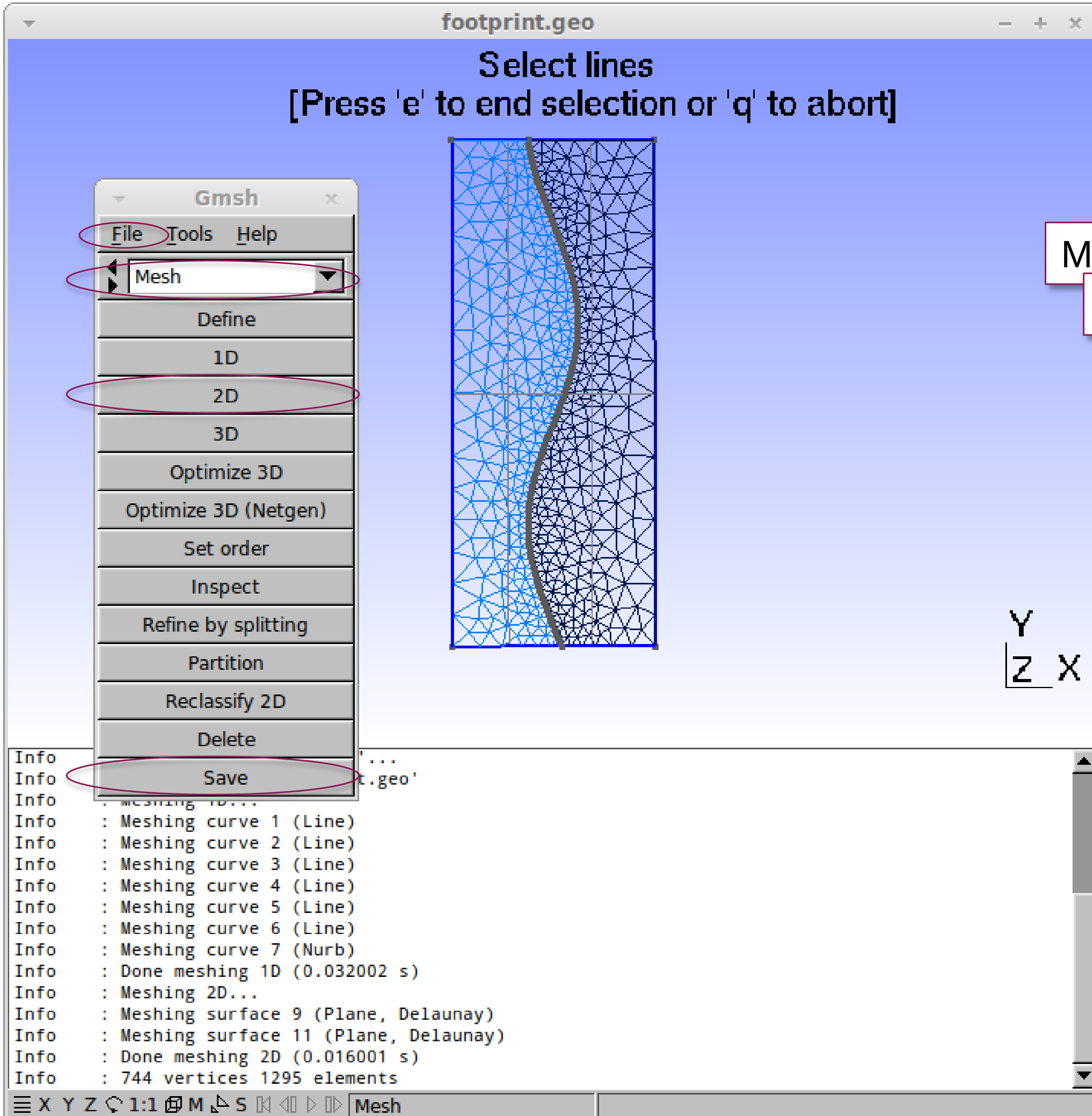
CSC

Physical groups

Add

Surface

Line



Mesh

2d

Save

File

Exit

Convert Mesh

- Conversion using `ElmerGrid`:

```
> ElmerGrid 14 2 footprint.msh
  -autoclean -order 1.0 0.1 0.01
```

- If you want to have a look in `ElmerPost`:

```
> ElmerGrid 14 3 footprint.msh
  -autoclean -order 1.0 0.1 0.01
```

External extrusion

- Extruding mesh using ExtrudeMesh
  - You can find it under `trunk/elmerice/Meshers/ExtrudeMesh.c`
- Extrusion done using DEM:

```
ExtrudeMesh footprint extruded_bounded
10 9.99 1 1 0 0.00 0.00 DEM 15.00 2.00 -9999.0 0
```

levels

Min. depth

partitions

baseline

#boundary layers

%height of BL

Ratio in BL

DEM directory

Cut-off radius

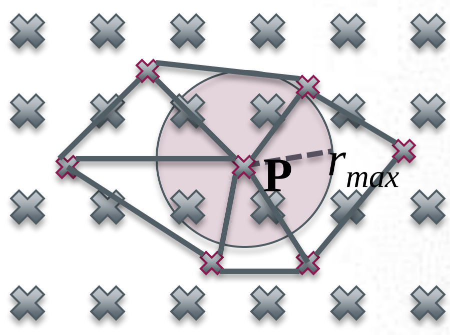
Exponent of inverse distance

NaN value

Correct bed

External extrusion

- Inverse distance



$$h(\mathbf{P}) = \left(\sum_{\|\mathbf{P}-\mathbf{X}_i\|<r_{\max}} h_i \left\|\mathbf{P} - \mathbf{X}_i\right\|^m \right) / \left(\sum_{\|\mathbf{P}-\mathbf{X}_i\|<r_{\max}} \left\|\mathbf{P} - \mathbf{X}_i\right\|^m \right)$$

Setup of run

- Initializing velocity field: **`diagnostic.sif`**
- Solvers:
  1. Depth: `ElmerIceSolvers FlowDepthSolver`
  2. Height: `ElmerIceSolvers FlowDepthSolver`
  3. Flow: `Equation= "NavierStokes"`
  4. Free Surface: `"FreeSurfaceSolver" "FreeSurfaceSolver"`
  5. Mesh: `Equation = "Mesh Update"`
  6. VTU (ParaView): `"ResultOutputSolve" "ResultOutputSolver"`

Never executed, as diagnostic

```
> ElmerSolver diagnostic.sif
```

Prognostic run

- New SIF:
- Switching to transient (time evolving)
  - Restart from previous solution
  - Enable mesh deformation
  - Need accumulation ablation function (MATC)

```
$ function accum(X) {\
  x0 = -
   sin(2.0*3.1415927*X(1)/2000.0)*125.0;\
  dx = abs(x0 - X(0));\
  if (X(2) > 500.0)\
    {_accum = 0.0;}\
  else\
    { netaccum = ((11.0/2750.0) * X(2) -
   0.85);\
      if (netaccum > 0.0)\
        { _accum = (1.0 - dx/250.0)*
   netaccum;}\
      else {_accum = netaccum;}\
    }\
}
```

```
> ElmerSolver prognostic.sif
```

Hydropotential

- Compile function:
  - `> elmerf90 SRC/GetHydroPotential.f90 -o GetHydroPotential.so`
- Run, by restarting (from post.result)
  - `> ElmerSolver hydrotest.sif`