Elmer GUI Tutorials

CSC – IT Center for Science

December 10, 2015

Elmer GUI Tutorials

About this document

The Elmer GUI Tutorials is part of the documentation of Elmer finite element software. Elmer GUI Tutorials gives examples on the use of Elmer in different field of continuum physics. Also coupled problems are included.

All these tutorials assume the use of ElmerGUI, the graphical user interface of Elmer. There are also older tutorials in the Elmer non-GUI Tutorials that may be used by advanced users.

The present manual corresponds to Elmer software version 8.1. Latest documentations and program versions of Elmer are available (or links are provided) at `http://www.csc.fi/elmer`.

Copyright information

Contents

Instructions for the GUI tutorials

Here are some instructions for following the GUI tutorials:

- All the needed input files should be available among the `ElmerGUI/samples` directory that should have come with the installation. Look under a subdirectory named after the suffix of the sample file.
- The instructions written in `verbatim` refer to operations with the GUI. Intendation means step in the menu hierarchy. The instructions should not be mixed with those in the command file.
- The menu structure for the default set of equations is located in directory `edf`, there are a few additional ones in directory `edf-extra`. These may be copied to the directory `edf` permanently, or be appended to the menus while running the ElmerGUI.
- The default menu structure may differ from the configuration used when writing the tutorial. Hence the user is encouraged to check by herself whether the menu structures exist or not.
- After having once defined the case you may go to the working directory and launch ElmerSolver from command-line. There you may edit the .sif file to alter the parameters.
- Manual alteration to the sif file will not be communicated to the ElmerGUI project. All editions will be overrun by the GUI when saving the project.
- ElmerPost is no longer available in all installations. In that case you may use VTK widget, or preferably Paraview with .vtu suffix on your output files.
- The cases have been run a number of times but errors are still possible. Reporting them to elmeradm@csc.fi, for example, is greatly appreciated.

Tutorial 1

Temperature distribution of an idealized geological intrusion

Directory: HeatIntrusion1D
Solvers: HeatSolve
Tools: ElmerGUI
Dimensions: 2D, Transient

Problem description

This is an extreme simplification of the intrusion process in related to geological processes. The case is effectively 1D but it is treated as 2D for generality of the approach.

We study temperature distribution from -8 km to -4 km. It is assumed that before the intrusion there is a linear temperature distribution from 320 to 160 C. At $t = 0$ an intrusion of 900 C replaces the temperature from -6.5 km to -5.5 km. The initial temperature distribution may therefore be presented by a piecewise linear function passing through the points depicted in table 1.1.

Table 1.1: Piecewise linear function defining the initial temperature

depth [m]	T (C)
-8000	320.0
-6500	260.0
-6499	900.0
-5501	900.0
-5500	220.0
-4000	160.0

The material parameters for the rock are assumed to be those of ideal granite. We take $k = 2.5$ W/mK, $C_p = 1250$ J/kgK, and $\rho = 2800$ kg/m^3.

As boundary conditions we use the initial temperature at the upper and lower end. Hence the problem depends only on the depth direction.

Solution procedure

Start `ElmerGUI` from command line or by clicking the icon in your desktop. Here we describe the essential steps in the ElmerGUI by writing out the clicking procedure. Tabulation generally means that the selections are done within the window chosen at the higher level.

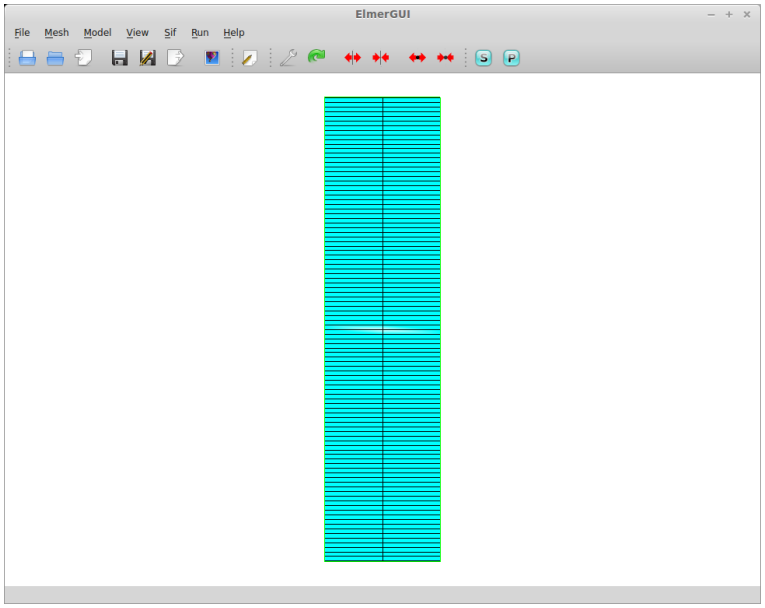

Figure 1.1: The geometry of the mesh in ElmerGUI

The mesh is given in ElmerGrid format in file `geoslab.grd` in the samples directory of ElmerGUI, load this file.

```
File
  Open -> geoslab.grd
```

You should obtain your mesh and may check in the `Model summary` window that it consists of 303 nodes and 200 quadrilateral surface elements. The mesh is very coarse in the $x-$direction intentionally since we are looking just for 1D solution. If the mesh was successfully imported your window should look something in figure 1.1.

After we have the mesh we start to go through the Model menu from the top to bottom. In the `Setup` we choose things related to the whole simulation such as file names, time stepping, constants etc. The simulation is carried out in 2-dimensional cartesian coordinates and in transient. We will use 1000 timesteps each of size 10 years. For time-stepping we will use 2nd order scheme. The results will be saved in `vtu` format for visualization with Paraview.

```
Model
  Setup
    Simulation Type = Steady transient
    Timestepping Method = BDF
    Timestepping Order = 2
    Timestep Intervals = 10000
    Timestep Sizes = $10*365*24*3600
    Output Intervals = 2
    Post File = case.vtu
```

Choose `Accept` to close the window.

In the equation section we choose the relevant equations and parameters related to their solution. In this case we'll have one set only one equation – the heat equation.

When defining Equations and Materials it is possible to assign the to bodies immediately, or to use mouse selection to assign them later. In this case we have just one body and one boundary and therefore its easier to assign the Equation and Material to it directly.

For the linear system solvers we are happy to use the defaults. One may however, try out different preconditioners (ILU1,...) or direct Umfpack solver, for example.

```
Model
  Equation
    Add
      Name = Heat Equation
      Apply to bodies = 1
      Heat Equation
        Active = on
  Apply
  OK
```

The Material section includes all the material parameters. They are divided to generic parameters which are direct properties of the material without making any assumptions on the physical model, such as the mass. Other properties assume a physical law, such heat conductivity.

```
Model
  Material
    Add
      Name = Granite
      Apply to bodies = 1
      General
        Density = 2700.0
        Heat Capacity = 1250.0
      Heat Equation
        Heat Conductivity = 2.5
    Apply
    OK
```

We will need an initial condition for the case.

```
Model
  Initial Condition
    Add
      Name = InitialState
      Heat Equation
        Temperature -> press enter and write the expression
      Apply to bodies = 1
    Apply
    OK
```

Now the expression to be written uses an internal linear interpolation feature of Elmer.

```
Variable coordinate 2
  Real
    -8000 320.0
    -6500 260.0
    -6499 500.0
    -5501 500.0
    -5500 220.0
    -4000 160.0
  End
```

In this case we only need boundary conditions for the upper and lower boundary. First we create the boundary conditions

```
Model
  BoundaryCondition
    Add
      Heat Equation
        Temperature = 320.0
      Name = Down
      OK
    Add
      Heat Equation
        Temperature = 160.0
      Name = Up
      OK
```

Then we set the boundary properties

```
Model
  Set boundary properties
```

Choose the defined group of three boundaries by clicking with the mouse and apply the condition for the lower boundary.

```
Boundary condition
  Down
```

and similarly for the upper boundary.

For the execution ElmerSolver needs the mesh files and the command file. We have know basically defined all the information for ElmerGUI to write the command file. After writing it we may also visually inspect the command file.

```
Sif
  Generate
  Edit... -> look how your command file came out
```

Before we can execute the solver we should save the files in a directory. In saving the project all the necessary files for restarting the case will be saved to the destination directory.

```
File
  Save Project
```

After we have successfully saved the files we may start the solver

```
Run
  Start solver
```

A convergence view automatically pops up showing relative changes of each iteration. As the case is linear only one iteration was required for the solution and the second one just is needed to check the convergence. The resulting output log is shown in figure 1.2.

Note: if you face problems in the solution phase and need to edit the setting, always remember to save the project before execution.

To view the results we use Paraview. The chosed vtu file may not be opened by Elmer tools.

```
Run
  Start ParaView
```

If your configuration is ok a Paraview window should pop up. Paraview may only open the 1st timestep. If this happens press `Open` and choose `case..vtu`. Choose temperature for the surface to be plotted. As we have a timeseries we may run the sequence by pressing the play icon.

You may also choose the `Plot Over Line` filter to see the data plotted over a line for better numerical inspection.

The final maximum temperature of the analysis is around 600 C. Whether this is close to correct could be studied by increasing further the time and space resolution.

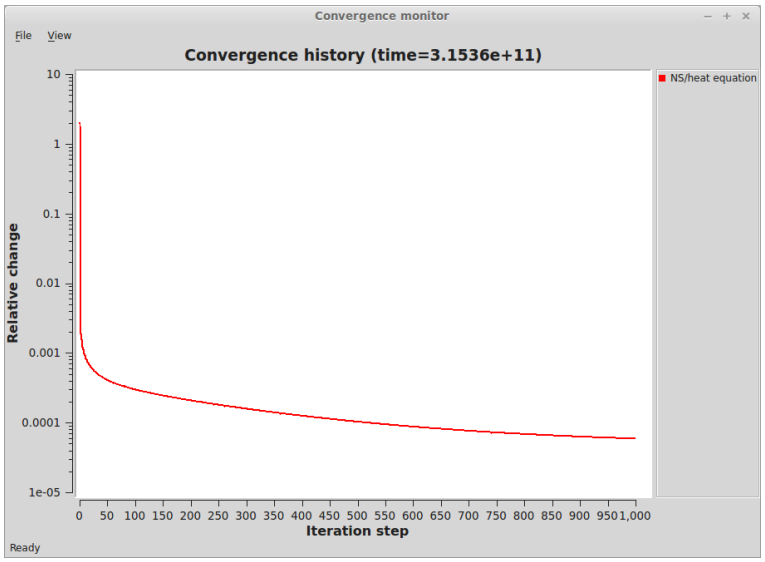

Figure 1.2: The output log of ElmerSolver when used under ElmerGUI

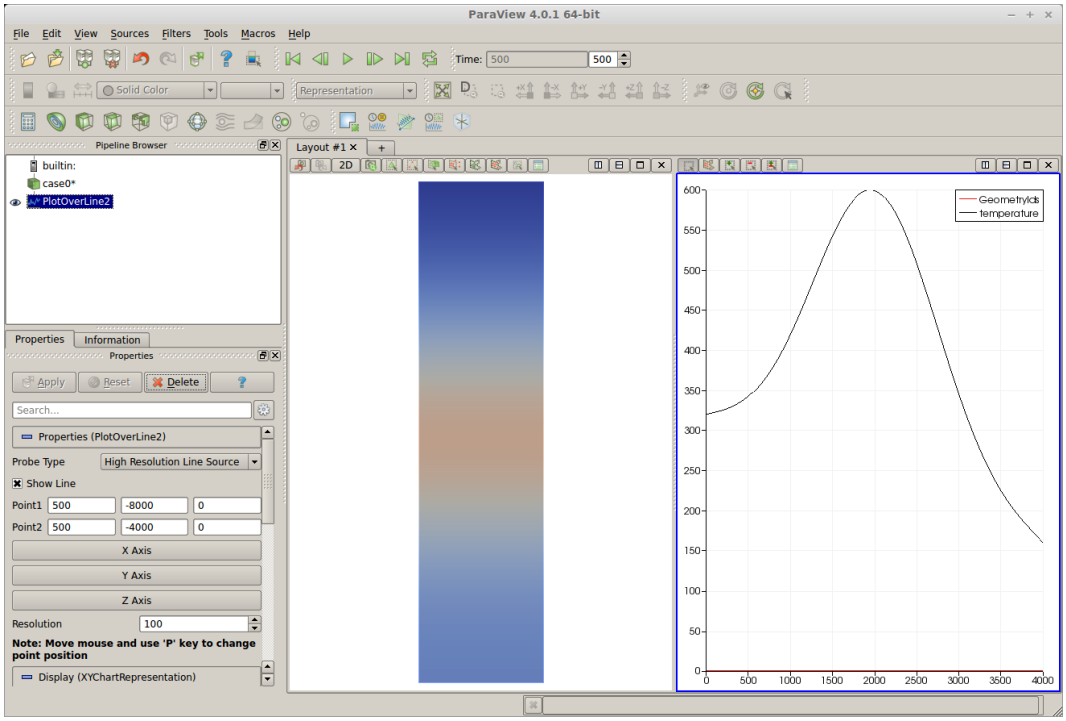

Figure 1.3: The data visualized in Paraview as a surface and lineplot

1.0.1 Variation accounting for latent heat release

For intrusion processes the latent heat often plays on important role. For that the internal phase change model can be used. It requires that all the internal energy is expressed using specific enthalphy rather than heat capacity and latent heat.

We now eliminate the heat capacity of the original case (1250 J/kgK) and create a specific enthalpy that includes heat capacity of 1000 J/kgJ and latent heat release of 200 kJ/kg release between interval 700–800 C.

The expression for `Specific Enthalpy` accounting for these two now yields

```
Variable Temperature
  Real
    0.0  0.0
    700  7.0e5
    800  9.0e5
    900  10.0e5
 End
```

For the phase change model we select

```
Model
  Equation
    Heat Equation
      Phase Change Model = spatial 2
  Update
  OK
```

With these changes the maximum temperature at the end of simulation cycle becomes around 624 C.

Tutorial 2

Heat equation – Temperature field of a solid object

Directory: TemperatureGenericGUI
Solvers: HeatSolve
Tools: ElmerGUI,netgen,OpenCascade
Dimensions: 3D, Steady-state

Problem description

This tutorial tried to demonstrate how to solve the heat equation for a generic 3D object. The solid object (see figure 2.1) is heated internally by a heat source. At some part of the boundary the temperature is fixed. Mathemetically the problem is described by the Poisson equation

$$\begin{cases} -\kappa \Delta T &= \rho f \quad \text{in} \quad \Omega \\ T &= 0 \quad \text{on} \quad \Gamma \end{cases} \tag{2.1}$$

where κ is the heat conductivity, T is the temperature and f is the heat source. It is assumed that density and heat conductivity are constants.

To determine the problem we assume that the part of the boundary is fixed at $T_0 = 293$ K, the internal heat generation is, $h = 0.01$ W/kg, and use the material properties of aluminium.

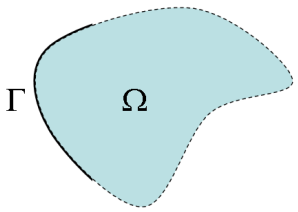

Figure 2.1: Generic object being heated

Solution procedure

Start `ElmerGUI` from command line or by clicking the icon in your desktop. Here we describe the essential steps in the ElmerGUI by writing out the clicking procedure. Tabulation generally means that the selections are done within the window chosen at the higher level.

The geometry is given in step format in file `pump_carter_sup.stp` in the `samples/step` directory of ElmerGUI, This file is kindly provided at the AIM@SHAPE Shape Repository by INRIA. The heat equation is ideally suited for the finite element method and the solution may be found even at meshes that for some other problems would not be feasible. Therefore you may easily experiment solving the same problem with different meshes. If you lack OpenCascade you might try to solve a similar problem with the `grd` files `angle3d.grd, angles3d.grd, bench.grd`, or `cooler.grd`, for example.

The CAD geometry defined by the step file is transformed on-the-fly by OpenCascade library into a stl file for which nglib creates tetrahedral volume discretization. You may also use the tetlib library (tetgen) if you have installed it as a plug-in.

Load the input file:

```
File
  Open -> pump_carter_sup.stp
```

The meshing will take a minute or two. You should obtain your mesh and may check in the number of element in the `Model summary`. With netgen the default setting generates 8371 nodes and 36820 tetrahedral elements. Visual inspection rewiels that the mesh is not quite satisfactory in geometric accuracy. We choose to modify the mesh by altering the settings in the following way.

```
View -> Cad model...
  Model -> Preferences...
    Restrict mesh size on surfaces by STL density = on
    Apply
Mesh -> Remesh
```

The meshing a take a minute or two. The modified mesh should include 16159 nodes and 65689 tetrahderal elements and be more appieling to the eye. In order to affect the mesh density study the command-line options of the netgen manual. Here we continue with the default mesh.

We want to set the temperature at the inside of the holes and in that aim you may join the three boundaries (see figure 2.2). For that aim we may choose the six pieces that constitute the boundaries as shown in the picture by pressing the `Ctrl`-key down.

```
Mesh
  Unify Surface
```

After we have the mesh we start to go through the Model menu from the top to bottom. In the `Setup` we choose things related to the whole simulation such as file names, time stepping, constants etc. The simulation is carried out in 3-dimensional cartesian coordinates and in steady-state. Only one steady-state iteration is needed as the case is linear.

```
Model
  Setup
    Simulation Type = Steady state
    Steady state max. iter = 1
```

Choose `Apply` to close the window.

In the equation section we choose the relevant equations and parameters related to their solution. In this case we'll have one set only one equation – the heat equation.

When defining Equations and Materials it is possible to assign the to bodies immediately, or to use mouse selection to assign them later. In this case we have just one body and therefore its easier to assign the Equation and Material to it directly, whereas the active boundary is chosen graphically.

For the linear system solvers we are happy to use the defaults. One may however, try out different preconditioners (ILU1,...), for example.

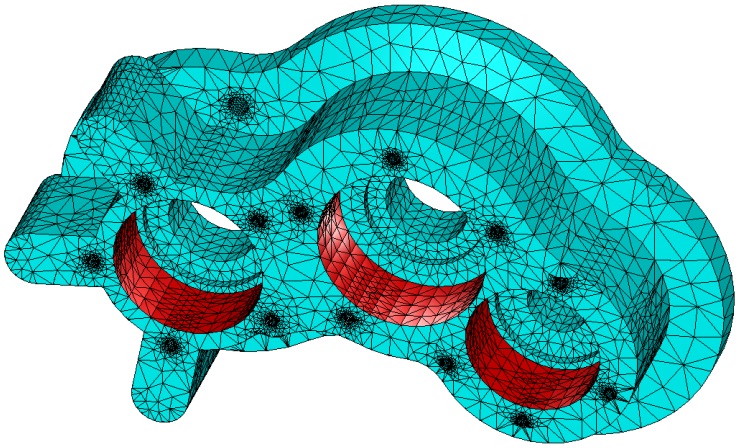

Figure 2.2: The computational mesh showing the three joined boundaries

```
Model
  Equation
    Add
      Name = Heat Equation
      Apply to bodies = Body 1
      Heat Equation
        Active = on
      Add
      OK
```

The Material section includes all the material parameters. They are divided to generic parameters which are direct properties of the material without making any assumptions on the physical model, such as the mass. Other properties assume a physical law, such heat conductivity. We choose Aluminium from the Material library which automatically sets for the needed material properties.

```
Model
  Material
    Add
      Material library
        Aluminium
      Apply to bodies = Body 1
      Add
      OK
```

A Body Force represents the right-hand-side of a equation that in this case represents the heat source.

```
Model
  Body Force
    Add
      Name = Heating
      Heat Source = 0.01
      Apply to bodies = Body 1
      Add
      OK
```

No initial conditions are required in steady state case.

In this case we have only one boundary and set it to room temperature. First we create the boundary condition

```
Model
  BoundaryCondition
    Add
      Heat Equation
        Temperature = 293.0
      Name = RoomTemp
      Add
      OK
```

Then we set the boundary properties

```
Model
  Set boundary properties
```

Choose the defined group of three boundaries by clicking with the mouse and apply the condition for this boundary.

```
Boundary condition
  RoomTemp
```

For the execution ElmerSolver needs the mesh files and the command file. We have know basically defined all the information for ElmerGUI to write the command file. After writing it we may also visually inspect the command file.

```
Sif
  Generate
  Edit -> look how your command file came out
```

Before we can execute the solver we should save the files in a directory. In saving the project all the necessary files for restarting the case will be saved to the destination directory.

```
File
  Save Project
```

After we have successfully saved the files we may start the solver

```
Run
  Start solver
```

A convergence view automatically pops up showing relative changes of each iteration. As the case is linear only one iteration was required for the solution and the second one just is needed to check the convergence. The norm of the solution should be around 432.4 K (with the default tetgen mesh 389.8 K, respectively).

Note: if you face problems in the solution phase and need to edit the setting, always remember to regenerate the sif file and save the project before execution.

Postprocessing

To view the results we may use the ElmerPost postprocessor or start the the internal VTK widget as is done here,

```
Run
  Postprocessor (VTK)
```

The default configuration shows just the object. To color the surface with the temperature choose

```
Surfaces
  Surface: Temperature
  Apply
```

The maximum temperature should be about 586.5 K. You may turn on opasity in order to see through the object, 10-20% is a good value. This way you'll able to see some isosurfaces that you might want to define. Some examples of the visualizations may be seen in figure 2.3.

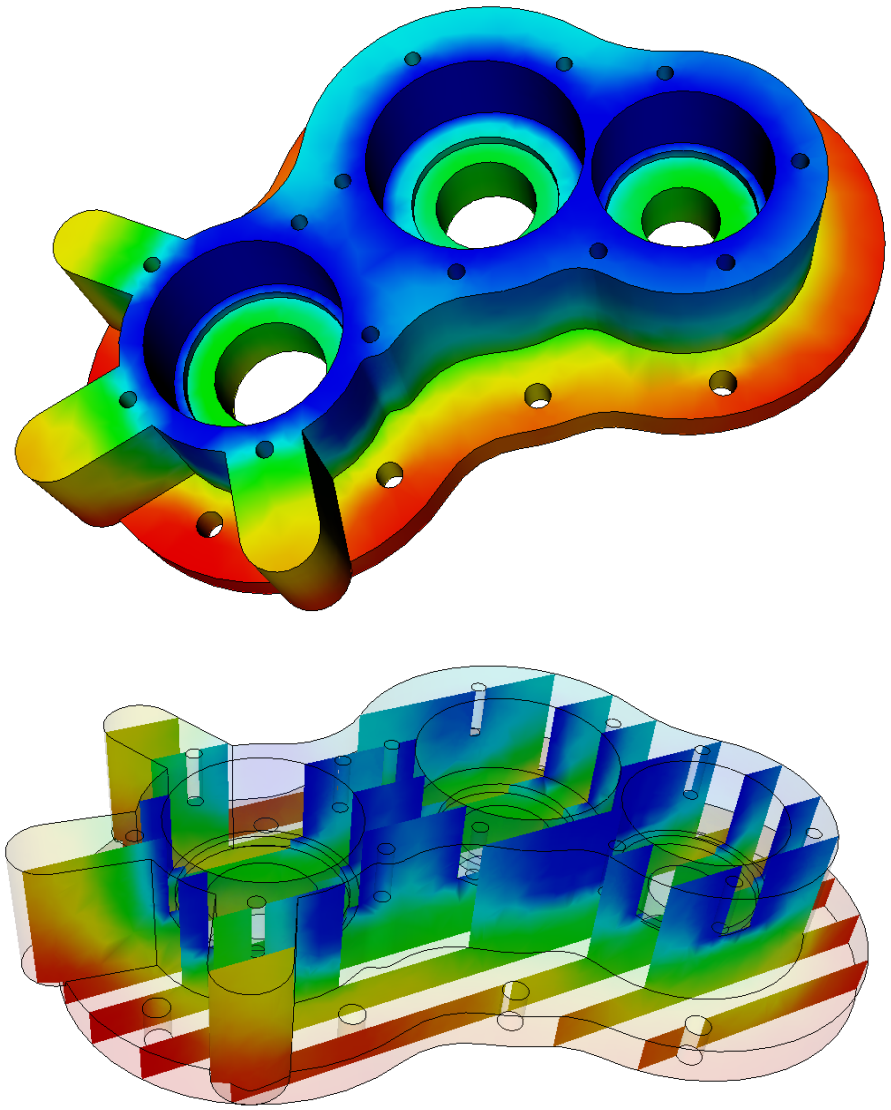

Figure 2.3: The temperature distribution of the solid object domain as visualized using the VTK-based postprocessor

Tutorial 3

Generic scalar PDE on 3D angle domain

Directory: ModelPDE3D
Solvers: ModelPDE
Tools: ElmerGUI
Dimensions: 3D, Steady-state

Problem description

This tutorial demonstrates the use of the generic advection-diffusion-reaction equation through ElmerGUI. The solver may be found in module ModelPDE. The purposes of the model pde and also this tutorial is to help those who want to understand Elmer from a mathematical perspective to be able to carry some own code development.

The problem is a simple 3d structure `angles.grd` that can be charectized by a $2 \times 2 \times 2$ topological grid where entries $(1,1,1)$, $(1,2,1)$, $(2,1,1)$ and $(2,1,2)$ are meshed. This is the simplest cartesian structure with full 3D solution.

We can rather freely play with the parameters of the Model PDE. The equation is generic and the parameters are assumed to be unit free. For detailed description of the problem see the description in Elmer Programmers Tutorial.

As a first suggestion, we will show how to make a simple case where the two extreme edges are set to zero using Dirichlet boundary conditions, and constant unity source term is applied to the body. This is the simple Poisson equation with constant coefficient.

Menu structures for Model PDE

The menu structures for the case are defined in `model-pde.xml`. If after starting you cannot find the menu structures add the file to the `edf` directory of your installation, or append the menu structures within ElmerGUI.
The following material parameters may be defined

`Diffusion Coefficient` `Real`
: Diffusion coefficient, μ.

`Reaction Coefficient` `Real`
: Reaction coefficient, λ.

`Time Derivative Coefficient` `Real`
: Multiplier of the time derivative, ρ.

`Convection Coefficient` `Real`
: Multiplier of convection coefficient, κ.

`Convection Velocity 1` `Real`
Convection velocity in direction x, a_x.

`Convection Velocity 2` `Real`
Convection velocity in direction y, a_y.

`Convection Velocity 3` `Real`
Convection velocity in direction z, a_z.

The following parameter defines the heat source f on the right-hand-side

`Field Source` `Real`

The menu structures defines the following parameters for boundary conditions:

`Field` `Real`
Dirichlet BC for the scalar field under study, u.

`Field Flux` `Real`
Neumann boundary condition for the field, q.

`Robin Coeffficient` `Real`

`External Field` `Real`
Coefficient α and external field value g for Robin boundary condition.

In transient cases the user may also give an initial condition for u,

`Field` `Real`

Solution procedure

Start `ElmerGUI` from command line or by clicking the icon in your desktop. Here we describe the essential steps in the ElmerGUI by writing out the clicking procedure. Tabulation generally means that the selections are done within the window chosen at the higher level.

The mesh is given in ElmerGrid format in file `angles.grd` in the samples directory of ElmerGUI, load this file.

```
File
  Open -> angles.grd
```

You should obtain your mesh and may check in the `Model summary` window that it consists of 35 721 nodes and 32 000 trilinear elements. If the mesh was successfully imported your window should look something in figure 3.1. The figure also shows the two extreme boundary patches that we intend to use Dirichlet conditions for.

After we have the mesh we start to go through the Model menu from the top to bottom. In the `Setup` we choose things related to the whole simulation such as file names, time stepping, constants etc. The simulation is carried out in 2-dimensional cartesian coordinates and in steady-state. Only one steady-state iteration is needed as the case is linear.

```
Model
  Setup
    Simulation Type = Steady state
    Steady state max. iter = 1
```

Choose `Accept` to close the window.

In the equation section we choose the relevant equations and parameters related to their solution. In this case we'll have one set only one equation – the Model PDE.

When defining Equations and Materials it is possible to assign the to bodies immediately, or to use mouse selection to assign them later. In this case we have just one body and one boundary and therefore its easier to assign the Equation and Material to it directly.

For the linear system solvers we are happy to use the defaults. One may however, try out different preconditioners (ILU1,...) or direct Umfpack solver, for example.

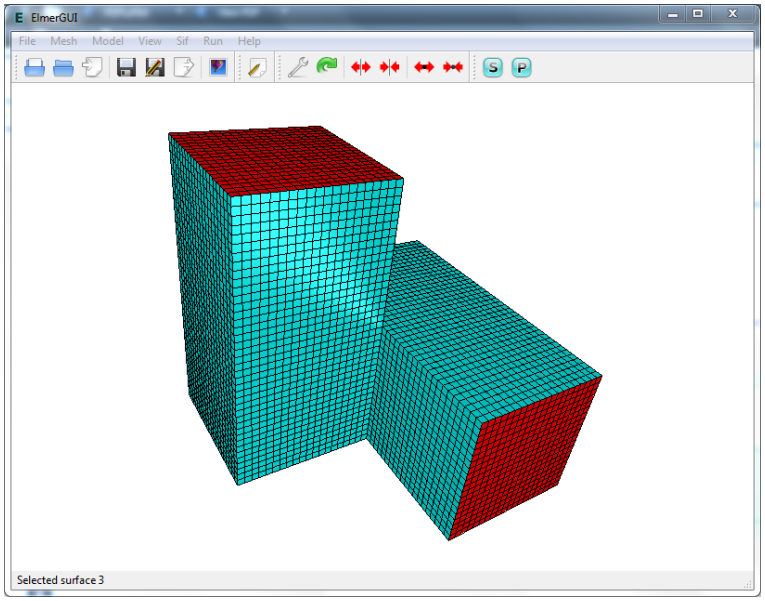

Figure 3.1: The finite element mesh in ElmerGUI

```
Model
  Equation
    Add
      Name = Model PDE
      Apply to bodies = 1
      Model PDE
        Active = on
  Apply
  OK
```

The Material section includes all the material parameters. If material parameter is not defined. It is assumed to be zero. Here we just set the diffusivity to one.

```
Model
  Material
    Add
      Name = Ideal
      Apply to bodies = 1
      Model PDE
        Diffusion Coefficient = 1.0
    Apply
    OK
```

A Body Force represents the right-hand-side of a equation,

```
Model
  Body Force
    Add
      Name = Source
      Field Source = 1.0
      Apply to bodies = 1
    Apply
    OK
```

No initial conditions are required in steady state case.

Finally, for the BCs first define them and then use the mouse to apply them to the correct boundary patches,

```
Model
  BoundaryCondition
    Add
      Model PDE
        Field = 0.0
      Name = Zero
    Apply
    OK
```

For the execution ElmerSolver needs the mesh files and the command file. We have know basically defined all the information for ElmerGUI to write the command file. After writing it we may also visually inspect the command file.

```
Sif
  Generate
  Edit -> look how your command file came out
```

Before we can execute the solver we should save the files in a directory. In saving the project all the necessary files for restarting the case will be saved to the destination directory.

```
File
  Save Project
```

After we have successfully saved the files we may start the solver

```
Run
  Start solver
```

A convergence view automatically pops up showing relative changes of each iteration. As the case is linear only one iteration was required for the solution and the second one just is needed to check the convergence.

The norm of the results at convergence should be 1.4243820.

Note: if you face problems in the solution phase and need to edit the setting, always remember to save the project before execution.

To view the results we may start the the internal VTK widget, or use ElmerPost for postprocessing. Also you may change the suffix of your `Post File` to `vtu` and perform the visualization using Paraview.

Further possibilities

Here are some things you could try out, or are at least possible. The menus of the GUI are just elementary so more advanced features may need that the keywords are added by hand to the command file. No coding is needed to implement these features though.

- Play with the reaction, diffusion, convection coefficient, and also the advection velocity. As far as they remain constant the equation should be solvable with one sweep.
- You could try to use Neumann or Robin BCs as well. Remember though that a steady state equatios needs definitions that uniquely define the solution.
- Make the problem time-dependent. Note that then you most likely need to define the coefficient for the time derivative.
- When the advection increases in size the solution may becore eventually oscillatory. To eliminate that some bubbles or p-elements may be needed. You may play around with element settings, for exammple use `Element = p:2` or `Element = n:1 b:1` etc.

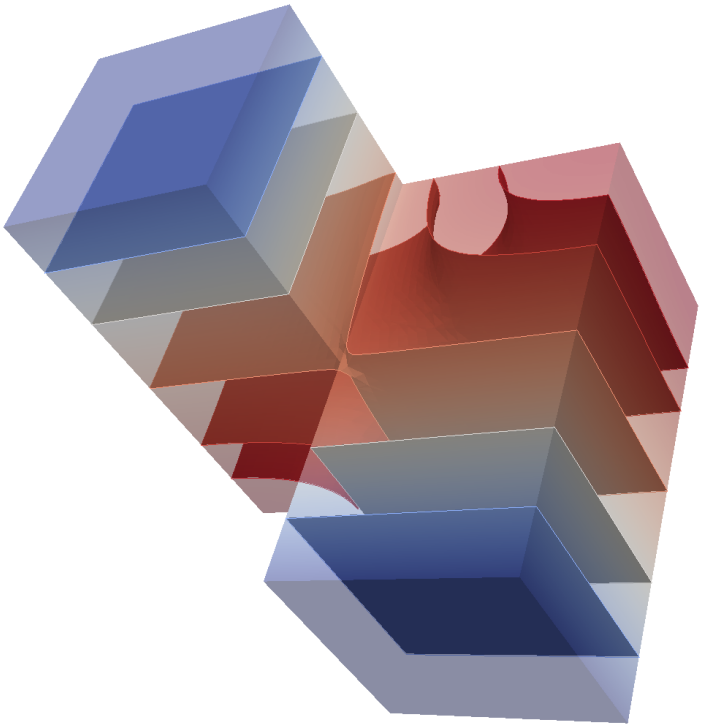

Figure 3.2: The field values of the structure as visualized with Paraview. Isosurfaces are defined for field values 0.5, 1.0, 1.5, 1.8 and 1.9, respectively.

- You could try to increase the number of elements either by using `-relh` parameter in ElmerGUI, or setting `Mesh Levels = 2` in Simulation section.
- You could try to use `MATC` to make the coefficients parameter dependent. Dependence on the solution itself introduces nonlinearities that might not be well handled by the fixed point iteration scheme. For more demanding nonlinearities Newton linearization or other technniques may be needed.
- Also some periodicity could be introduced to this problem by letting the two extreme surface patches have a dependence between them.
- You could introduce sort of contact conditions for the surface or bulk values of the problem by defining minimum or maximum values for the field.

Tutorial 4

Linear elasticity equation – Loaded elastic beam

Directory: ElasticBeam3D
Solvers: StressSolve
Tools: ElmerGUI
Dimensions: 3D, Steady-state

Case definition

Assume a homogenous, elastic beam being rigidly supported on one end. On the other end it is subjected with a load of 2000 N resulting from an attached object in the gravitational field. The gravity affects also the beam itself. The length of the beam is 1 m and the thickness is 0.05 m, and the width 0.1 m. Material properties of the beam are those of dry pine timber: Poisson ratio 0.37, Young's modulus $10 \cdot 10^9 \mathrm{N/m}^2$, and density 550 kg/m^3. The problem is to solve the displacement and stress field of the beam. Here the `StressSolve` routine based on the linear theory of elasticity is applied.

Solution procedure

The mesh is given in ElmerGrid format in file `beam3d.grd`, load this file.

```
File
  Open -> beam3d.grd
```

You should obtain your mesh and may check that it consists of 6073 nodes and of 1200 quadratic hexahedral elements. The second order elements give improved accuracy compared to the first order elements as they avoid the phenomenom known as locking.

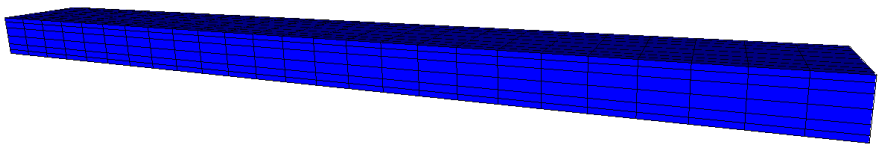

Figure 4.1: The mesh used in the computations

After we have the mesh we start to go through the Model menu from the top to bottom. In the Setup we choose things related to the whole simulation such as file names, time stepping, constants etc. The simulation is carried in steady-state in 3-dimensional cartesian coordinates.

```
Model
  Setup
    Simulation Type = Steady state
    Steady state max. iter = 1
```

In the Equation section we choose the relevant equations which in this case only includes the `Linear elasticity` equation which solves the problem according to linear elastic theory. We also want to compute the stresses as a post-processing step. For the linear system solvers we change the default settings in order to obtain a better convergence in this case. As the equation is fully linear we also eliminate the nonlinear iteration loop.

```
Model
  Equation
    Name = Elasticity
    Apply to Bodies = Body 1
    Linear elasticity
      Active = on
      Calculate Stresses = on
    Edit Solver Setting
      Linear System
        Method = Iterative / GCR
        Preconditioning = ILU1
      Nonlinear system
        Max. iterations = 1
      Apply
    Add
    OK
```

The Material section includes all the material parameters. They are divided to generic parameters which are direct properties of the material without making any assumptions on the physical model, such as the mass. Other properties assume a physical law, such as Young's modulus and Poisson ratio.

```
Model
  Material
    Name = Pine
    General
      Density = 550
    Linear Elasticity
      Youngs Modulus = 10.0e9
      Poisson ratio = 0.37
    Apply to Bodies = Body 1
    Add
    OK
```

In this case there is a body force i.e. the gravity acting on the beam. We assume that the gravity points to the negative y direction.

```
Model
  BodyForce
    Name = Gravity
    Linear Elasticity
      Force 2 = $ -9.81 * 550
    Apply to Bodies = Body 1
```

```
    Add
    OK
```

Here we use a MATC expression for computing the volume force. This expression is constant and is computed when the command file is interpreted.

Convergence should be obtained with the default initial condition i.e. zero for all fields, hence no initial condition is applied.

The first boundary condition fixes the beam rigidly at the wall. The second boundary condition distributes the load of 2000 N uniformly on the area of 5.0e-3 m^2.

```
Model
  BoundaryCondition
    Name = Wall
    Linear elasticity
      Displacement 1 = 0.0
      Displacement 2 = 0.0
      Displacement 3 = 0.0
    Add
    New

    Name = Mass
    Linear elasticity
      Force 2 = -4.0e5
    Add
```

The conditions may also be assigned to boundaries in the Boundary condition menu, or by clicking with the mouse. Here we use the latter approach as that spares us of the need to know the indexes of each boundary.

```
Model
  Set boundary properties
    Choose the wall end of the beam -> set boundary condition Wall
    Choose the other end of the beam -> set boundary condition Mass
```

For the execution ElmerSolver needs the mesh files and the command file. We have know basically defined all the information for ElmerGUI to write the command file. After writing it we may also visually inspect the command file.

```
Sif
  Generate
  Edit -> look how your command file came out
```

Before we can execute the solver we should save the files in a directory. The project includes all the files needed to restart the case.

```
File
  Save Project
```

After we have successfully saved the files we may start the solver

```
Run
  Start solver
```

The simulation may take a minute or so depending on the speed of the processor. This time the convergence monitor does not have a meaningfull output since the of the different steps only one is related to the actual solution and the six other ones to the computation of stresses with the Galerkin method.

Results

When there are some results to view we may start the postprocessor, this time we use ElmerPost.

```
Run
  Start postprocessor
```

As a result the absolute value of maximum displacement is shown. The maximum displacement is 6.36 cm To visualize the displacement in the geometry using ElmerPost can be done with the following command in the `Elmer-Post` command line.

```
  math n0=nodes
  math nodes=n0+Displacement
```

To redraw the picture with new settings use the rightenmost icon on the top row. The resulting picture is shown in Fig 4.2 Note that the displacement are so large that the assumption of linearity may be severely

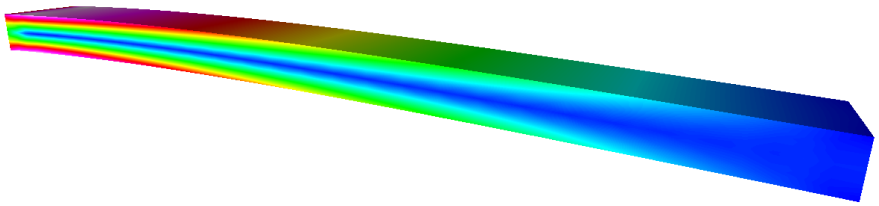

Figure 4.2: The displaced shape of the elastic beam colored with the von Mises stresses

questioned. When further increasing the loading one should resort to a solver that is able to catch the geometric nonlinearities.

Extra task: Gravity in x direction

The beam should be more rigid if the beam is oriented differently. For that aim, change the direction of gravity to orient in the negitive x. Change the body force

```
Model
  BodyForce
    Linear Elasticity
      Force 1 = $ -9.81*550
    Update
    OK
```

and the boundary condition

```
Model
  BoundaryCondition
    Linear elasticity
      Force 1 = -4.0e5
    Update
    OK
```

The rigidity should scale as dh^3 and hence the maximum displacement should be reduced roughly to one quarter of the original.

Tutorial 5

Navier-Stokes equation – Laminar incompressible flow passing a step

Directory: FlowStepGUI
Solvers: FlowSolve
Tools: ElmerGUI
Dimensions: 2D, Steady-state

Case definition

This tutorial represents the canonical step flow of viscous fluid. A fluid, flowing past a step (see figure 5.1), has the density 1 kg/m and viscosity 0.01 kg/ms. The velocity profile at the inlet is parabolic with a mean velocity $< v_x >= 1.0$ m/s and $v_y = 0.0$ m/s. At the outlet only the vertical component is defined, $v_y = 0.0$ m/s. At all other walls the no-slip boundary condition, $\vec{v} = 0$, is applied. Thus the Reynolds number for the case is around 100.

Figure 5.1: Geometry of the step flow problem

Mathematically the problem to be solved is

$$\begin{cases} -\nabla \cdot (2\mu\bar{\bar{\varepsilon}}) + \rho\vec{u} \cdot \nabla\vec{u} + \nabla p &= 0 \quad \text{in } \Omega \\ \nabla \cdot \vec{u} &= 0 \quad \text{in } \Omega \end{cases} \tag{5.1}$$

with the boundary conditions

$$\begin{cases} u_x &= 1 \quad \text{on } \Gamma_{inlet} \\ u_x &= 0 \quad \text{on } \Gamma_{no-slip} \\ u_y &= 0 \quad \text{on } \Gamma_{inlet} \cup \Gamma_{outlet} \cup \Gamma_{no-slip} \end{cases} \tag{5.2}$$

where μ is the viscosity, $\bar{\bar{\varepsilon}}$ is the strain tensor, ρ is the density, $\vec{u}$ is the velocity and p is the pressure. It is assumed that the density and viscosity are constants.

Solution procedure

The mesh is given in ElmerGrid format in file `step.grd`, load this file.

```
File
  Open -> step.grd
```

You should obtain your mesh and may check that it consists of 9696 nodes and of 9442 bilinear elements.

```
Model
  Summary...
```

After we have the mesh we start to go through the Model menu from the top to bottom. In the Setup we choose things related to the whole simulation. The steady-state simulation is carried out in 2-dimensional cartesian coordinates, which are also the defaults.

```
Model
  Setup
    Simulation Type = Steady state
    Coordinate system = Cartesian
```

In the equation section we choose the relevant equations and parameters related to their solution. In this case the only the Navier-Stokes equation is needed.

When defining Equations and Materials it is possible to assign the to bodies immediately, or to use mouse selection to assign them later. In this case we have just one body and therefore its easier to assign the Equation and Material to it directly. One could also edit the solver setting in order to try different strategies for solving the nonlinear or linear system. Initially the Navier-Stokes solver uses the more robust Picard iteration which is changed to Newton iteration after few initial steps. For the given viscosity the default values are ok, but may need tuning when going into higher Reynolds numbers.

```
Model
  Equation
    Name = Navier-Stokes
    Apply to Bodies = Body 1
    Navier-Stokes
      Active = on
      Edit Solver Setting
        Nonlinear System
          Max. iterations = 20
          Newton after iterations = 3
    Add
    OK
```

The Material section includes all the material parameters. They are divided to generic parameters which are direct properties of the material without making any assumptions on the physical model, such as the density. Other properties assume a physical law, such as viscosity.

```
Model
  Material
    Name = Ideal
    General
      Density = 1.0
    Navier-Stokes
      Viscosity = 0.01
    Apply to Bodies = Body 1
    Add
    OK
```

The current case does not have any body forces. Convergence should also be obtained using the default initial condition which sets all field values to zero. Hence no setting for initial condition are needed.

Only one boundary condition may be applied to each boundary and therefore all the different physical BCs for a boundary should be grouped together. In this case the Temperature and Velocity. The side walls are assumed to be adiabatic.

The parabolic inlet-profile is achieved using the MATC environment. To be able to edit the content of the inlet profile click `Enter` to open an edit box for the `Velocity 1`. The given expression will be interpreted at run-time so that $v_x = 6(y-1)(2-y)$. As $y \in [1,2]$ thereby creating a parabolic velocity profile with a mean velocity of unity.

```
Model
  BoundaryCondition
    Name = Inlet
    Navier-Stokes
      Velocity 1 = Variable Coordinate 2; Real MATC "6*(tx-1)*(2-tx)"
      Velocity 2 = 0.0
    Add
    New

    Name = Outlet
    Navier-Stokes
      Velocity 2 = 0.0
    Add
    New

    Name = Walls
    Navier-Stokes
      Noslip wall BC = on
    Add
    OK
```

The conditions may also be assigned to boundaries in the Boundary condition menu, or by clicking with the mouse. Here we use the latter approach as that spares us of the need to know the indexes of each boundary.

```
Model
  Set boundary properties
    Choose Inlet -> set boundary condition Inlet
    Choose Outlet -> set boundary condition Outlet
    Choose Walls -> set boundary condition Walls
```

For the execution ElmerSolver needs the mesh files and the command file. We have now basically defined all the information for ElmerGUI to write the command file. After writing it we may also visually inspect the command file.

```
Sif
  Generate
  Edit -> look how your command file came out
```

Before we can execute the solver we should save the files in a directory. The project includes all the files needed to restart the case. Create a suitable directory for the case if needed.

```
File
  Save Project
```

After we have successfully saved the files we may start the solver

```
Run
  Start solver
```

A convergence view automatically pops up showing relative changes of each iteration. The problem should converge in about ten iterations to a norm of 0.4347 visible on the output.

When there are some results to view we may start the postprocessor also

```
Run
  Start postprocessor
```

Results

The results may be viewed using the postprocessor as shown in Figure 5.2 and 5.3. One may also register specific values, for example the pressure difference is 0.388 Pa, the minimum and maximum lateral velocities are -0.1666 m/s and 1.5 m/s, respectively. One special result of interest is the point, on the x-axis, at which the direction of the flow changes. In this case its position is about 5.0 m after the step.

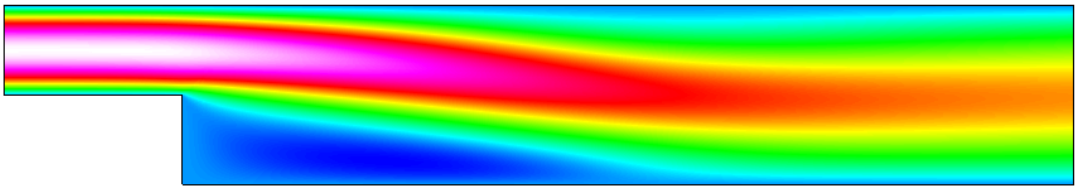

Figure 5.2: Absolute value of the velocity field

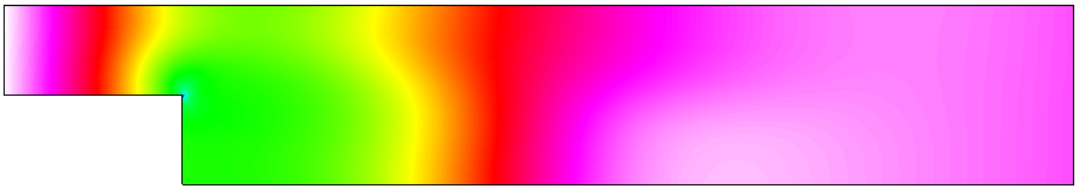

Figure 5.3: Pressure field

Extra task: Decreasing the viscosity

Try what happens if the viscosity is further decaresed by a factor 10. Convergence may be difficult to obtain. Some tricks that may be tested include

- Introducting a relaxation factor (typically in the range 0.5–0.7)
- Increasing number of nonlinear iterations
- Favoring Picard iteration over Newton
- Increasing mesh density (and length of domain)

Don't be worried if you fail to find convergence. This task will mainly act as a motivator in using turbulence models for higher Reynolds numbers.

Remember to re-perform the following phases in order to get the updated results

```
Sif
  Generate
File
  Save Project
Run
  Start solver
```

You may just reload the results in the postprocessor rather than closing and opening the program.

Tutorial 6

Transient flow and heat equations – Rayleigh-Benard instability

Directory: RayleighBenardGUI
Solvers: HeatSolve, FlowSolve
Tools: ElmerGUI
Dimensions: 2D, Transient

Case definition

This tutorial is about simulating the developing of the Rayleigh-Benard instability in a rectangular domain (Figure 6.1) of dimensions 0.01 m height and 0.06 m length. The simulation is performed with water and the material parameters of water required by the Elmer model are presented in Table 6.1. The temperature difference between the upper and lower boundary is set to 0.5 so that lower one has the temperature of 293.5 K and the upper one has the temperature of 293 K.

The density of water is inversely proportional to its temperature. Thus, heated water starts to flow upwards, and colder downwards due to gravity. In this case we assume that the Boussinesq approximation is valid for thermal incompressible fluid flow. In other words, the density of the term $\rho\vec{f}$ in the incompressible Navier-Stokes equation can be redefined by the Boussinesq approximation

$$\rho = \rho_0(1 - \beta(T - T_0))$$

where β is the heat expansion coefficient and the subscript 0 refers to a reference sate.

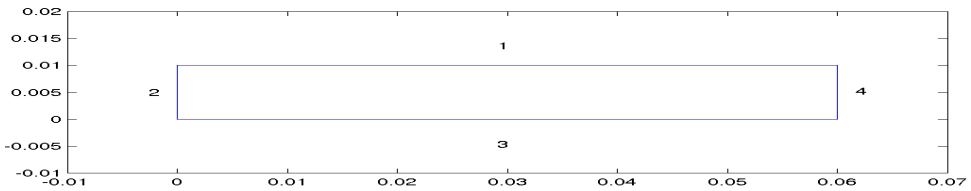

Figure 6.1: Domain.

Table 6.1: Material parameters for water

parameter	value
density	998.3 kg/m^3
viscosity	1040e-6 Ns/m^2
heat capacity	4183 J/(kg·K)
heat conductivity	0.58 W/(m·K)
heat expansion coefficient	2.07e-4 K^{-1}
reference temperature	293 K

Solution procedure

The mesh is given in ElmerGrid format in file `rectangle.grd`, load this file.

```
File
  Open -> rectangle.grd
```

You should obtain your mesh and may check that it consists of 3036 bilinear elements.

There is a possibility to divide and unify edges to simplify the case definition in the future.

```
Choose (left wall + right wall (Ctrl down)) -> unify edge
```

After we have the mesh we start to go through the Model menu from the top to bottom. In the Setup we choose things related to the whole simulation such as file names, time stepping, constants etc. The simulation is carried out in 2-dimensional cartesian coordinates. 2nd order bdf time-stepping method is selected with 200 steps and with step size of two seconds. Gravity is needed for the bouyancy force and it is defined by a vector with fourc components. The three define a unit vector and the fourth its magnitude.

```
Model
  Setup
    Simulation Type = Transient
    Steady state max. iter = 20
    Time Stepping Method = bdf
    BDF Order = 2
    Time Step Intervals = 200
    Time Step Sizes = 2.0
    Gravity = 0 -1 0 9.82
```

In the equation section we choose the relevant equations and parameters related to their solution. In this case we'll have one set of equations (named "Natural Convection") which consists of the heat equation and of the Navier-Stokes equation.

When defining Equations and Materials it is possible to assign the to bodies immediately, or to use mouse selection to assign them later. In this case we have just one body and therefore its easier to assign the Equation and Material to it directly. It is important to select the convection to be computed since that couples the velocity field to the heat equation.

The system may include nonlinear iterations of each equation and steady state iterations to obtain convergence of the coupled system. It is often a good idea to keep the number of nonlinear iterations in a coupled case low. Here we select just one nonlinear iteration for both equations. For the linear system solvers we are happy to use the defaults. One may however, try out different preconditioners (ILU1,...) or direct Umfpack solver, for example.

```
Model
  Equation
    Name = Natural Convection
    Apply to Bodies = 1
```

```
    Heat Equation
      Active = on
      Convection = Computed
      Edit Solver Setting
        Nonlinear System
          Max. iterations = 1
    Navier-Stokes
      Active = on
      Edit Solver Setting
        Nonlinear System
          Max. iterations = 1
    Add
    OK
```

The Material section includes all the material parameters. They are divided to generic parameters which are direct properties of the material without making any assumptions on the physical model, such as the mass. Other properties assume a physical law, such as conductivities and viscosity.

Here we choose water at room temperature from the material library. You may click trough the material parameters of the various solvers to ensure that the properties are indeed as they should be. Any consistant set of units may be used in Elmer. The natural choice is of course to perform the computations in SI units.

Apart from the properties from the material database, we reference temperature for the Boussinesq approximation.

```
Model
  Material
    Material library
      Water (room temperature)
    General
      Reference Temperature = 293
    Apply to Bodies = 1
    Add
    OK
```

A Body Force represents the right-hand-side of a equation. It is generally not a required field for a body. In this case, however, we apply the buoyancy resulting from heat expansion as a body force to the Navier-Stokes equation.

```
Model
  Body Force
    Name = Buoyancy
    Apply to Bodies = 1
    Navier-Stokes
      Boussinesq = on
    Add
    OK
```

Initial conditions should be given to transient cases. In this case we choose a constant Temperature field and an small initial velocity that initializes the symmetry break.

```
Model
  Initial Condition
    Name = Initial Guess
    Heat Equation
      Temperature = 293
    Navier-Stokes
      Velocity 1 = 1.0e-9
      Velocity 2 = 0.0
```

Only one boundary condition may be applied to each boundary and therefore all the different physical BCs for a boundary should be grouped together. In this case the Temperature and Velocity. The side walls are assumed to be adiabatic.

```
Model
  BoundaryCondition
    Name = Bottom
    Heat Equation
      Temperature = 293.5
    Navier-Stokes
      Velocity 1 = 0.0
      Velocity 2 = 0.0
    Add
    New

    Name = Top
    Heat Equation
      Temperature = 293
    Navier-Stokes
      Velocity 1 = 0.0
      Velocity 2 = 0.0
    Add
    New

    Name = Sides
    Navier-Stokes
      Velocity 1 = 0.0
      Velocity 2 = 0.0
    Add
```

The conditions may also be assigned to boundaries in the Boundary condition menu, or by clicking with the mouse. Here we use the latter approach as that spares us of the need to know the indexes of each boundary.

```
Model
  Set boundary properties
    Choose Bottom -> set boundary condition Bottom
    Choose Top -> set boundary condition Top
    Choose Sides -> set boundary condition Sides
```

For the execution ElmerSolver needs the mesh files and the command file. We have know basically defined all the information for ElmerGUI to write the command file. After writing it we may also visually inspect the command file.

```
Sif
  Generate
  Edit -> look how your command file came out
```

Before we can execute the solver we should save the files in a directory. The project includes all the files needed to restart the case.

```
File
  Save Project
```

After we have successfully saved the files we may start the solver

```
Run
  Start solver
```

A convergence view automatically pops up showing relative changes of each iteration.

When there are some results to view we may start the postprocessor also

```
Run
  Start postprocessor
```

Results

Due to the number of the time-steps the simulation may take around ten minutes. You may inspect the results with ElmerPost as the time-steps are computed, or wait until all timesteps have been computed. When opening the result file using ElmerGUI ElmerPost only opens the first time-step. Therefore it is important to reopen the file and load the time-steps of interest. Pressing the button `All` selects all the calculated time steps. A video of the results can be viewed by selecting the option `Timestep Control` and pressing the button `Loop` under the `Edit` menu.

In Figures 6.2 and 6.3 the obtained temperature distribution and the velocity vectors are presented. The maximum velocity in the system should be about 0.516 mm/s.

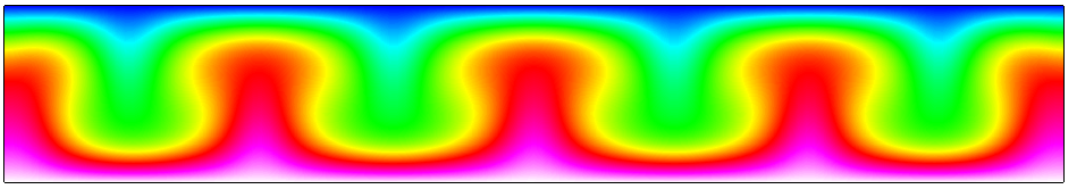

Figure 6.2: Temperature distribution at 260 s.

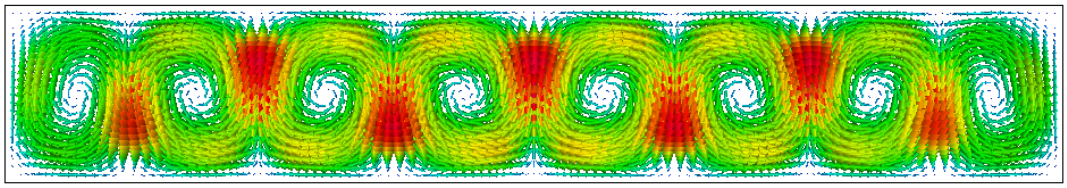

Figure 6.3: Velocity vectors at 260 s.

Extra task: Sensitivity to temperature difference

If you have time you may try to solve the case with different parameters. Changing the temperature difference is one way of affecting the instability of the system. Decreasing the tempereture differences the system eventually becomes steady state and the convection rolls vanish alltogether. Increasing the temperature difference may increase the number of convection rolls and eventually the system becomes fully chaotic. Note that changing the temperature difference also affects to the time scale of the wake.

Tutorial 7

Interaction between fluid flow and elastic obstacle

Directory: FsiObstacleGUI
Solvers: FlowSolve,ElasticSolve,MeshSolve
Tools: ElmerGUI
Dimensions: 2D, Steady-state

Case definition

This tutorial demonstrates how to set up a coupled case of fluid-structure interaction. Flow is initiated at one end of a channel which has an elastic obstacle that bends under the influence of fluidic forces. The resulting displacements modify the domain thereby affecting the flow of the fluid.

The channel is assumed to be 2D. The length of the channel is 10 m and the height is 2 m. At the distance of 2 m from the inlet sits an elastic beam with a rounded top the height of which is 1.2 m and width 0.4 m. A parabolic velocity profile with maximum velocity of 1 m/s is assumed.

Material properties are assumed to be rather simple: For the structure the density is 1000 kg/m^3, Youngs modulus is 1000 Pa, and Poisson ratio 0.3. For the fluid the density is 1 kg/m^3 and viscosity is 0.1 Pas. Additionally the fluid has elastic properties that are used to extend the displacement of the elastic beam to the fluid.

The idea of the case definition is to demonstrate simple yet strongly coupled FSI case without getting into turbulence modeling. Realistic material parameters for the given size of the domain would easily result to turbulence and just small displacements.

The case is solved using standard weak coupling with some relaxation to boost the convergence. The solution is steady-state so only the final results are will be studied.

Solution procedure

The nonlinear elasticity equation is not by default active in the menu structures ElmerGUI. Hence, the user must load these before starting the simulations.

```
File
  Definitions
    Append -> nonlinearelastity.xml
```

The additional definitions should reside in the directory `edf-extra` within the distribution. Moving the desired `xml` files to the `edf`-directory enables automatic loading of the definitions at start-up. By inspecting the definitions in the `Elmer Definitions File editor` one may inspect that the new definitions were really appended.

The mesh is defined in .in2d format, the 2D format of netgen, in file `obstacle_in_channel.in2d`, load this file.

```
File
  Open -> obstacle_in_channel.in2d
```

The default mesh is obviously too sparse. To make the mesh more dense set

```
Mesh -> Configure -> nglib -> Max H: 0.1
```

and choose

```
Mesh -> Remesh
```

You should obtain a denser mesh and may check that it consists of around 4140 nodes and 7890 linear triangles.

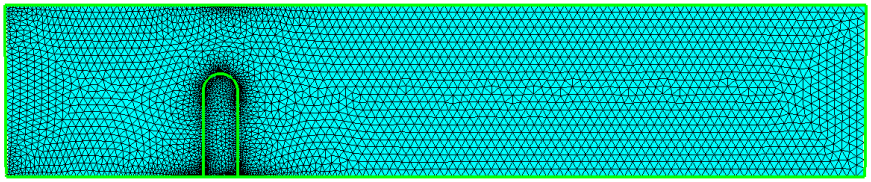

Figure 7.1: The mesh of the obstacle in channel case as seen in ElmerGUI

After we have the mesh we start to go through the `Model` menu from the top to bottom. In the `Setup` we choose things related to the whole simulation such as file names, time stepping, constants etc. The simulation is carried out in 2-dimensional cartesian coordinates in steady-state. There is not much to do here, just increase the number of iterations needed for the convergence of the coupled system. We also set the output interval to zero which means that results are written only at the end of the case.

```
Model
  Setup
    Coordinate system = Cartesian
    Simulation type = Steady state
    Steady state max. iter = 100
    Output Intervals = 0
    ...
```

In the Equation section we choose the relevant equations and parameters related to their solution. In this case we'll have two different sets of solvers (called as Equation in Elmer slang). The fluid domain consists of flow and mesh deformation solvers, while the elastic domain just includes the nonlinear elasticity solver. We'll name them appropriately.

To enhance the convergence and efficient use of resources we set relaxation factors of the primary solvers to 0.5 and the number of nonlinear iterations to 1. The mesh deformation solver just extents the displacements of the elasticity solver to the fluid domain, so no relaxation is needed here. For the linear systems we are quite happy with the defaults.

To honor the causality the flow solver should be solved first, then the elasticity solver and as last the mesh deformation solver. We set the priorities accordingly.
The equation for the fluid flow + mesh deformation

```
Model
  Equation
    Add
    Name = Flow and mesh deform
```

```
    Apply to Bodies = 1
    Navier-Stokes
      Active = on
      Priority = 2
      Edit Solver Setting
        Nonlinear System
          Max.iterations = 1
          Nonlinear System Relaxation Factor = 0.5
    Mesh Update
      Active = on
      Priority = 0
    OK
```

and then for the solid

```
Model
  Equation
    Add
    Name = Elasticty
    Apply to Bodies = 2
    Nonlinear Elasticty
      Active = on
      Priority = 1
      Edit Solver Setting
        Nonlinear System
          Max.iterations = 1
          Nonlinear System Relaxation Factor = 0.5
    OK
```

Next we set our rather simple material parameters. The Material section includes all the material parameters. They are divided into generic parameters which are direct properties of the material without making any assumptions on the physical model, such as the density. Other properties assume a physical law, such as conductivities and viscosity.

```
Model
  Material
    Add
    Name = Ideal fluid
    General
      Density = 1.0
    Navier-Stokes
      Viscosity = 0.1
    Mesh Update
      Elastic Modulus = 1.0
      Poisson Ratio = 0.3
    Apply to Bodies = 1
    OK

    Add
    Name = Ideal structure
    General
      Density = 1.0e3
    Nonlinear Elasticity
      Youngs Modulus = 1.0e3
      Poisson Ratio = 0.3
    Apply to Bodies = 2
    OK
```

The `Body force` section usually represents the right-hand-side of an equation. In this case we do not need any body forces.

Also an `Initial condition` could be given in steady-state case to enhance convergence. However, in this case convergence is pretty robust with the default guess of zero.

We have five different boundary conditions: inflow, outflow, lateral walls with no-slip conditions, fsi conditions, and the beam base. As it is tedious to know the indexes by heart we first define the different BCs and only afterwards apply them to the existing boundaries with the mouse.

```
Model
  BoundaryCondition
    Name = Inflow
    Navier-Stokes
      Velocity 1 = Variable Coordinate 2
        Real MATC "tx*(2-tx)"
      Velocity 2 = 0.0
      Mesh Update 1 = 0.0
    Add
    New
```

The condition for `Velocity 1` above may easiest be typed by pressing `Enter`-key in the edit box which will open a larger window for editing.

```
    Name = Outflow
    Navier-Stokes
      Velocity 2 = 0.0
    Mesh Update
      Mesh Update 1 = 0.0
    Add
    New

    Name = Walls
    Navier-Stokes
      NoSlip Wall BC = on
    Mesh Update
      Mesh Update 1 = 0.0
      Mesh Update 2 = 0.0
    Add
    New

    Name = Base
    Nonlinear Elasticity
      Displacement 1 = 0.0
      Displacement 2 = 0.0
    Add
    New
```

The essense of fluid-structure interaction is in the following boundary condition. When the `Fsi BC` is active the fluidic forces are automatically within ElasticSolver. The backcoupling to Navier-Stokes is achieved through the change in fluid geometry which is enforced by the conditions for the `MeshSolver`.

```
    Name = FSI
    Nonlinear Elasticity
      FSI BC = on
    Navier-Stokes
      NoSlip Wall BC = on
    Mesh Update 1 = Equals Displacement 1
    Mesh Update 2 = Equals Displacement 2
```

Now we are ready to choose the boundaries

```
Model
  Set boundary properties
    Choose inlet side -> set boundary condition Inflow
    Choose outlet side -> set boundary condition Outflow
    Choose upper and lower sides (three pieces) -> set boundary condition Walls
    Choose obstacle base -> set boundary condition Base
    Choose interface between fluid and solid (two pieces) -> set boundary condition FS
```

For the execution ElmerSolver needs the mesh files and the command file. We have know basically defined all the information for ElmerGUI to write the command file. After writing it we may also visually inspect the command file.

```
Sif
  Generate
  Edit -> look how your command file came out
```

Before we can execute the solver we should save the files in a directory. The project includes all the files needed to restart the case. It's a good idea to give the project an illuminating name. Avoid paths which includes empty spaces since they may cause problems later on.

```
File
  Save Project
    Make New Folder -> fsi_obstacle
    OK
```

After we have successfully saved the files we may start the solver

```
Run
  Start solver
```

A convergence view automatically pops up showing relative changes of each iteration. The simulation may take around 10 seconds depending on your platform.

The computed norms should be around 0.514 for the Navier-Stokes solver, 0.108 for the elasticity solver, and 0.0548 for the mesh update solver. These are reached after 18 iterations using the rather strict default settings.

If there is some discrepancy the setup of the system was probably different from the one in the tutorial. If the results are in agreement we are ready to look at the results.

Results

To visualize the results open the postprocessor, in this case `ElmerPost` After the simulation has terminated we may open the postprocessor to view the results.

```
Run
  Start postprocessor
```

A standard way of visualizing is to choose `ColorMesh` and there choose `Surface` or `Both` and the desired field variable, for example `Velocity_abs` or `Pressure`. The mesh deformation is not active in all output formats. To activate the deformation in ElmerPost you may enter the following sequence of command to the command line at the bottom of ElmerPost window.

```
math n0=nodes
math nodes=n0+Displacement
```

The maximum speed in the system is around 2.328 and the maximum displacement 0.2636. Note that for the saved results the displacement and mesh update fields have been merged into one. In Figures 7.2, 7.3, and 7.4 the obtained velocity, pressure and displacement distributions are presented in the deformed mesh.

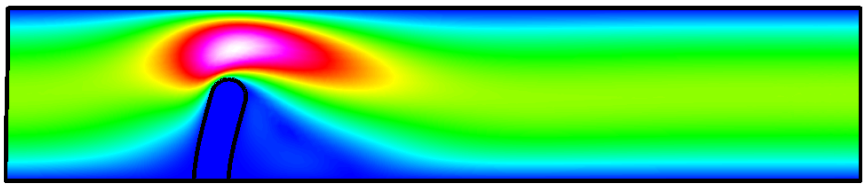

Figure 7.2: Velocity distribution of the obstacle in channel case.

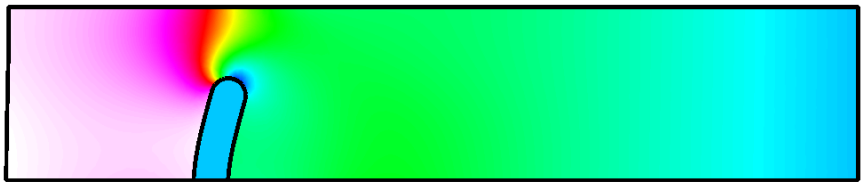

Figure 7.3: Pressure distribution of the obstacle in channel case.

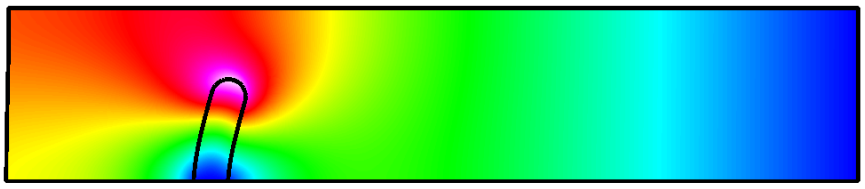

Figure 7.4: Displacement distribution of the obstacle in channel case.