ElmerGUI manual v. 0.2

Mikko Lyly

October 29, 2008

Contents

1 Introduction

ElmerGUI is a graphical user interface for the Elmer software suite [1]. The program is capable of importing finite element mesh files in various formats, generating finite element partitionings for models with piecewise linear boundaries, setting up PDE-systems to solve, and exporting model data and results for ElmerSolver and ElmerPost. There is also an internal postprocessor, which can be used to plot color surfaces, contours, vector fields, and visualize time dependent data.

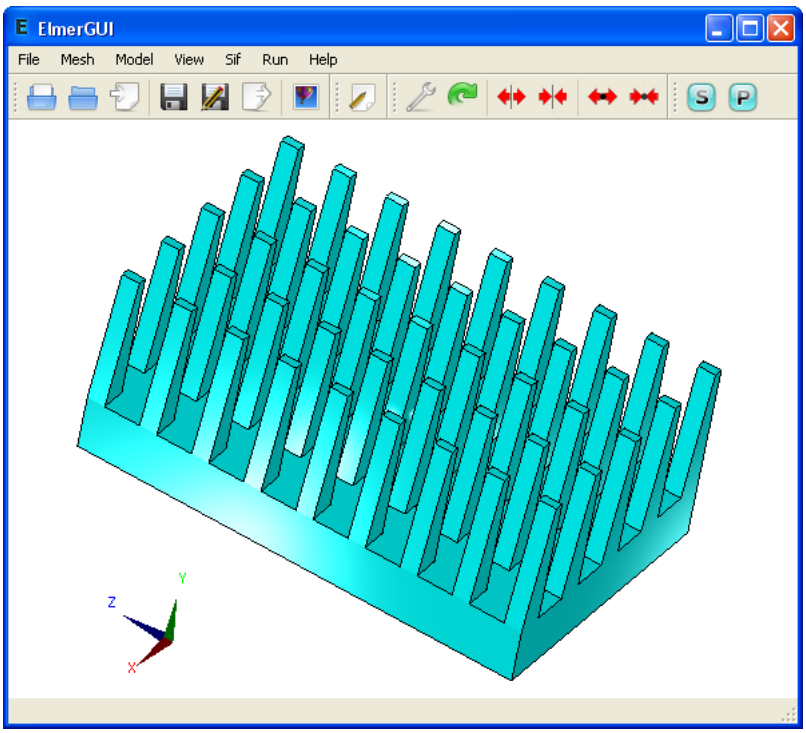

Figure 1: Main window of ElmerGUI.

One of the main features of ElmerGUI is the interface to the parallel solver engine. The GUI hides from the user a number of operations that are normally performed from command line with various external tools related to domain decomposition, launching the paralle processes, and merging the results. This makes it possible to use ElmerSolver with multi-core processors even on interactive desktop environments.

The menus of ElmerGUI are programmable and it should be relatively easy to strip and customize the interface for an proprietary application. An example of customizing the menus is provided in appendix A.

ElmerGUI relies on the Qt4 cross platform framework [3] and it uses the Qwt5 libray [4] for technical applications to monitor the solution. The internal postprocessor is based on VTK (Visualization Toolkit) [6]. The program is capable of using Tetgen [5] and Netgen [2] as external finite element mesh generators.

2 Installation from source

The source code of ElmerGUI is available from the subversion repository of SourceForge.Net. The GPL licenced source may be downloaded by executing the following command with a SVN client program (on Windows the Tortoise SVN client is recommended):

```
svn co https://elmerfem.svn.sourceforge.net/svnroot/elmerfem/trunk trunk
```

This will retrieve the current development version of the Elmer-suite.

Bofore staring to compile, please make sure that you have the development packages of Qt4 and Qwt5 installed on your system (i.e., libraries, headers, and program development tools). Qt version 4.2 or higher is recommended. If you want to compile the internal postprocessor, then the development packages of VTK version 5 or higher are also required.

The program is compiled by executing the following sequence of commands in a terminal window:

```
$ cd elmerfem/trunk/ElmerGUI
$ qmake
$ make
```

It is possible that the project file "ElmerGUI.pro" needs to be edited depending on how and where the external libraries have been installed. The lines that need attention can be found from the beginning of the file.

Finally, once the build process has finished, it suffices to set up the environment variable `ELMERGUI_HOME`, add it to `PATH`, and copy the file "ElmerGUI" and the directory "edf" in this location:

```
$ export ELMERGUI_HOME=$ELMER_HOME/bin
$ export PATH=$PATH:$ELMERGUI_HOME
$ cp -r ./ElmerGUI ./edf $ELMERGUI_HOME
```

Later, it is possible to upgrade ElmerGUI by typing

```
$ svn update
$ make
```

and by copying the new executable as well as the updated edf directory in `ELMERGUI_HOME`.

3 Input files

3.1 Geometry input files and mesh generation

ElmerGUI is capable of importing finite element mesh files and generating two or three dimensional finite element partitionings for bounded domains with piecewise linear boundaries. It is possible to use one of the following mesh generators:

- ElmerGrid (built-in)
- Tetgen (optional)
- Netgen (optional)

The default import filter and mesh generator is ElmerGrid. Tetgen and Netgen are optional modules, which may or may not be available depending on the installation.

An import filter or a mesh generator is selected automatically by ElmerGUI when a geometry input file is opened from the File menu:

File → Open...

The selection is based on the input file suffix according to Table 1. If two or more generators are capable of handing the same format, then the user defined "preferred generator" will be used. The preferred generator is defined in

Mesh → Configure...

Suffix	ElmerGrid	Tetgen	Netgen
.FDNEUT	yes	no	no
.grd	yes	no	no
.msh	yes	no	no
.mphtxt	yes	no	no
.off	no	yes	no
.ply	no	yes	no
.poly	no	yes	no
.smesh	no	yes	no
.stl	no	yes	yes
.unv	no	yes	no

Table 1. Input files and capabilities of the mesh generators.

Once the input file has been opened, it is possible to modify the mesh parameters and remesh the geometry. The mesh parameters can be found from the same place as the preferred generator. The control string for Tetgen has been discussed and explained in detail in [5].

The mesh generator is reactivated from the Mesh menu by choosing

Mesh → Remesh

In case of problems, the meshing thread may be terminated from

Mesh → Terminate meshing

3.2 Elmer mesh files

An Elmer mesh consists of the following four text files (detailed description of the file format can be found from Appendix B):

```
mesh.header
mesh.nodes
mesh.elements
mesh.boundary
```

The files have to be located side by side in the same mesh directory.

Elmer mesh files may be loaded and/or saved by opening the mesh directory from the File menu:

File → Load mesh...

and/or

File → Save as...

3.3 Project files

An ElmerGUI project consists of a project directory containing Elmer mesh files and a xml-formatted document `egproject.xml` describing the current state and settings. Projects may be loaded and/or saved from the File menu as

File → Load project...

and/or

File → Save project...

When an ElmerGUI project is loaded, a new solver input file will be generated and saved in the project directory using the sif-name defined in

Model → Setup...

If there is an old solver input file with the same name, it will be overwritten.

The contents of a typical project directory is the following:

```
case.sif
egproject.xml
ELMERSOLVER_STARTINFO
mesh.boundary
mesh.elements
mesh.header
mesh.nodes
```

4 Model definitions

4.1 Setup menu

The general setup menu can be found from

Model → *Setup...*

This menu defines the basic variables for the "Header", "Simulation", and "Constants" blocks for a solver input file. The contents of these blocks have been discussed in detail in the SolverManual of Elmer [1].

4.2 Equation menu

The first "dynamical menu" constructed from the ElmerGUI definition files (see Appendix A) is

Model → *Equation*

This menu defines the PDE-system to be solved as well as the numerical methods and parameters used in the solution. It will be used to generate the "Solver" blocks in a solver input file.

A PDE-system (a.k.a "Equation") is defined by choosing

Model → *Equation* → *Add...*

Go through the tabs and check the "Active" boxes of all individual equations that constitute your model. The numerical methods and parameters can be selected and tuned by pressing the "Edit Solver Settings" button. Finally, name the PDE-system in the "Name" line edit box, and apply the system to a set of bodies. Once the PDE-system has been defined, press the Ok-button. The equation remains visible and editable under the Model menu.

Tip: It is possible to attach an equation to a body by holding down the SHIFT-key while double clicking one of its surfaces. A pop up menu will then

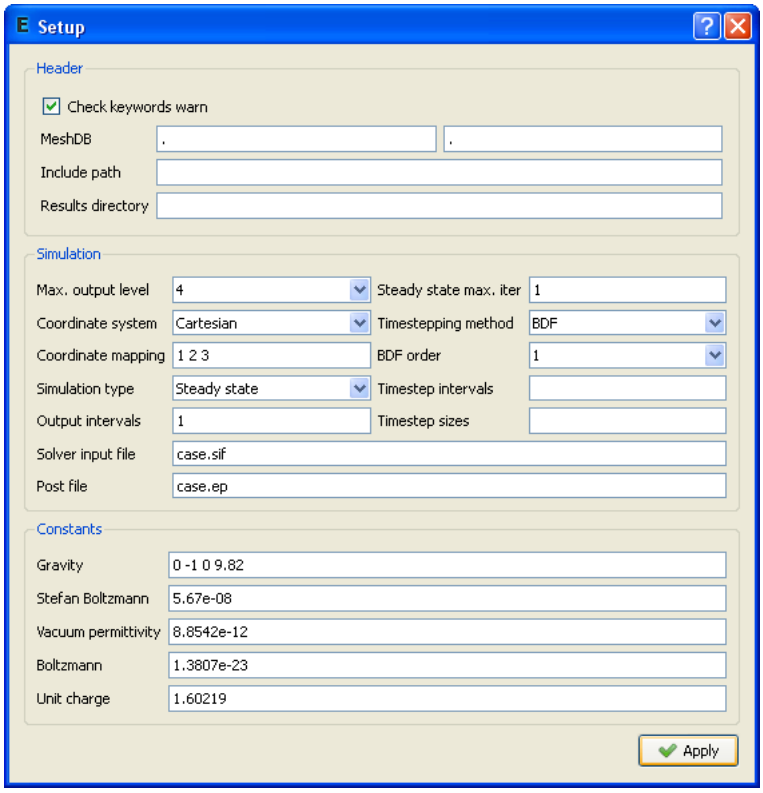

Figure 2: Setup menu.

appear, listing all possible attributes that can be attached to the selection. Choose the attributes and press Ok.

4.3 Material menu

The next menu is related to material and model parameters:

Model → *Material*

This menu will be used to generate the "Material" blocks in a solver input file.

In order to define a material parameter set and attach it to bodies, choose

Model → *Material* → *Add...*

Again, it is possible to attach the material to a body by holding down the SHIFT-key while double clicking one of its boundaries.

Note: The value of density should always be defined in the "General"

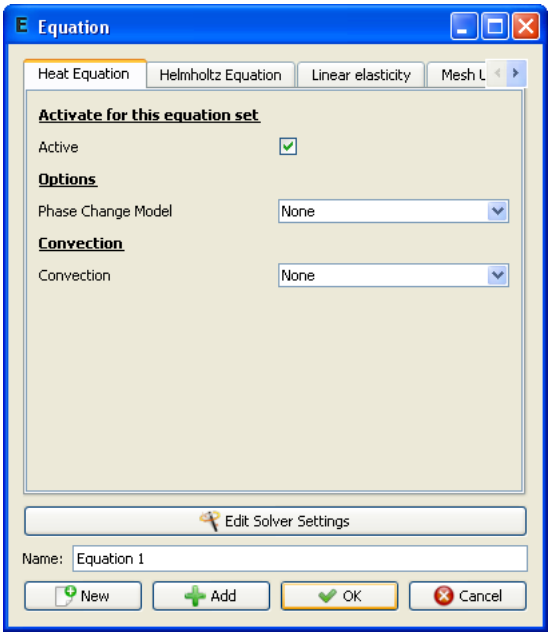

Figure 3: Equation menu.

tab. This field should never be left undefined.

Tip: If you set focus in a line edit box of a dynamical menu and press Enter, a small text edit dialog will pop up. This allows the input of more complicated expressions than just constants. As an example, go to *Model* → *Material* and choose *Add...* Place the cursor in the "Heat conductivity" line edit box of "Heat equation" and press Enter. You can then define the heat conductivity as a function of temperature as a piecewise linear function. An example is show in Figure N. In this case, the heat conductivity gets value 10 if the temperature is less than 273 degrees. It then rises from 10 to 20 between 273 and 373 degrees, and remains constant 20 above 373 degrees.

Tip: If the user presses SHIFT and F1, a tooltip for the active widget will be displayed.

4.4 Body force menu

The next menu in the list is

Model → *Body force*

This menu is used to construct the "Body force" blocks in a solver input file.

Again, choose

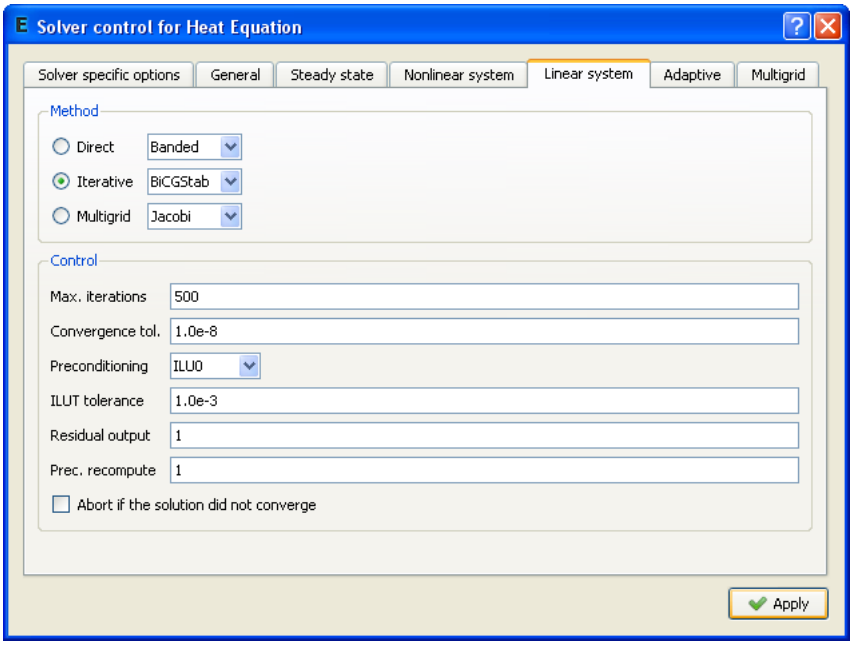

Figure 4: Solver settings menu.

Model → *Body force* → *Add...*

to define a set of body forces and attach it to the bodies.

4.5 Initial condition menu

The last menu related to body properties is

Model → *Initial condition*

Once again, choose

Model → *Initial condition* → *Add...*

to define a set of initial conditions and attach it to the bodies.

This menu is used to construct the "Initial condition" blocks in a solver input file.

4.6 Boundary condition menu

Finally, there is a menu entry for setting up the boundary conditions:

Model → *Boundary condition*

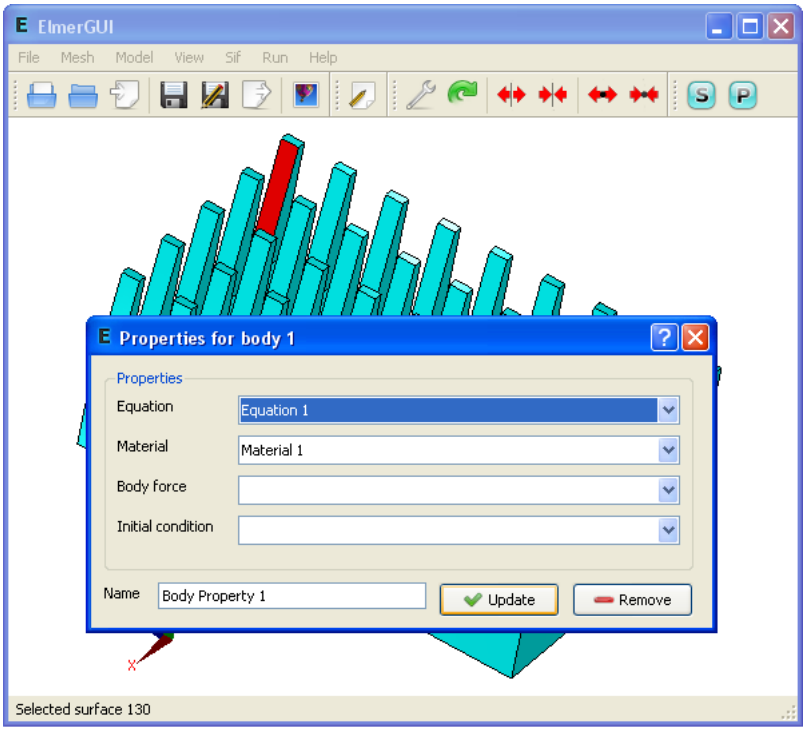

Figure 5: Body property editor is activated by holding down the SHIFT key while double clicking a surface.

Choose

Model → *Boundary condition* → *Add...*

to define a set of boundary conditions and attach them to boundaries.

Tip: It is possible to attach a boundary condition to a boundary by holding down the ALT or ALTGR-key while double clicking a surface or edge. A pop up menu will appear, listing all possible conditions that can be attached to the selection. Choose a condition from the combo box and finally press Ok.

5 Utility functions

5.1 Boundary division and unification

Some of the input file formats listed in Table 1 are not necessarily best suited for FE-calculations. The .stl format (stereo litography format), for example, is unable to distinguish between different boundary parts with different at-

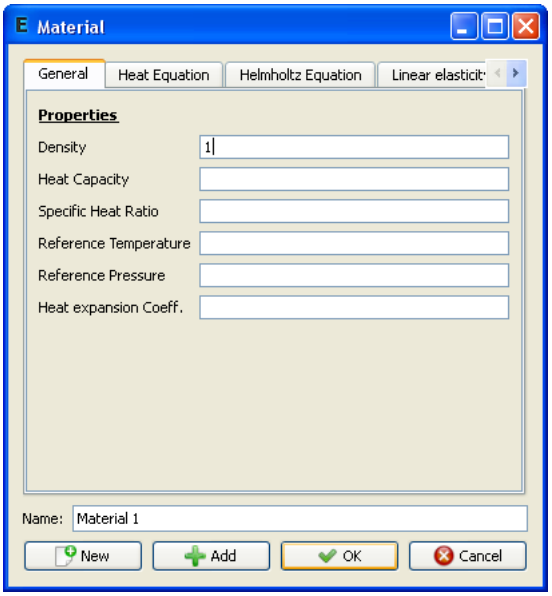

Figure 6: Material menu.

tributes. Moreover, it approximates the boundary by disconnected triangles. The file format would be rather useless in this context, unless there were other ways to identify the boundary parts and methods for dealing with the possible discontinuities.

ElmerGUI provides a minimal set of tools for boundary division and unification. The division is based on "sharp edge detection". An edge between two boundary elements is considered sharp, if the angle between the normals exceeds a certain value (20 degrees by default). The sharp edges are then used as a mortar to divide the surface into parts. The user may perform a sharp edge detection and boundary division from the Mesh menu by choosing

Mesh → *Divide surface...*

In 2D the corresponding operation is

Mesh → *Divide edge...*

The resulting parts are enumerated starting from the first free index.

Sometimes, the above process produces too many distinct parts, which eventually need to be (re)unified. This can be done by selecting a group of surfaces by holding down the CTRL-key while double clicking the surfaces and choosing

Mesh → *Unify surface...*

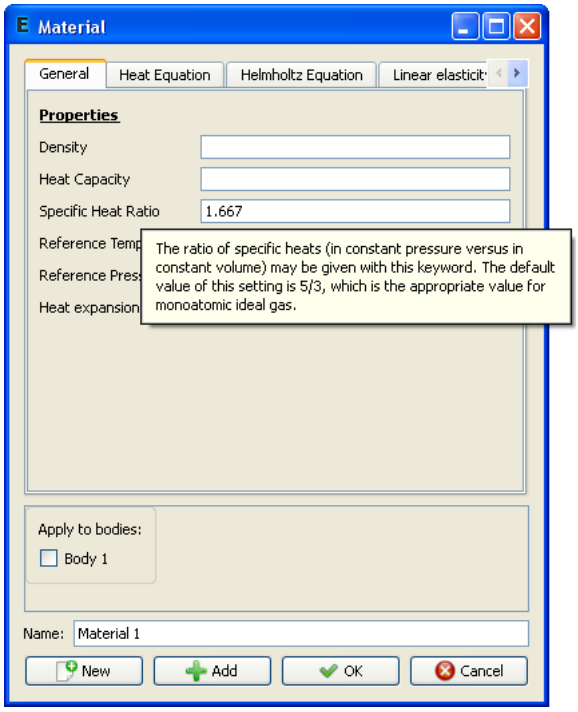

Figure 7: Tooltips are shown by holding down the SHIFT and F1 keys.

The same operation in 2D is

Mesh → *Unify edge...*

The result will inherit the smallest index from the selected group.

The sharp edges that do not belong to a closed loop may be removed by

Mesh → *Clean up*

This operation has no effect on the boundary division, but sometimes it makes the result look better.

5.2 Saving pictures

The model drawn on the display area may be scanned into a 24-bit RGB image and saved in several picture file formats:

File → *Save picture as...*

The function supports .bmp, .jpg, .png, .pbm, .pgm, and .ppm file extensions.

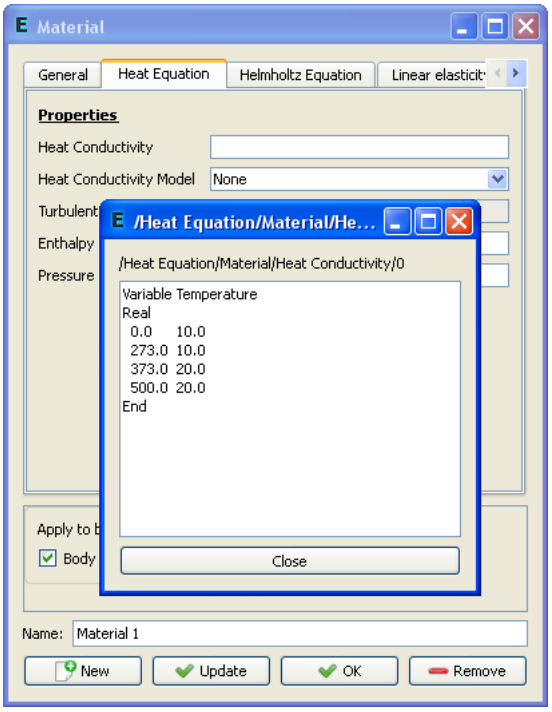

Figure 8: Text edit extension of a line edit box is activated by pressing Enter.

5.3 View menu

The View menu provides several utility functions for controlling the visual behaviour of ElmerGUI. The function names should be more or less self explanatory.

6 Solver input files

The contents of the Model menu are passed to the solver in the form of a solver input file. A solver input file is generated by choosing

Sif → *Generate*

The contents of the file are editable:

Sif → *Edit...*

The new sif file needs to saved before it becomes active. The recommended method is

File → *Save project...*

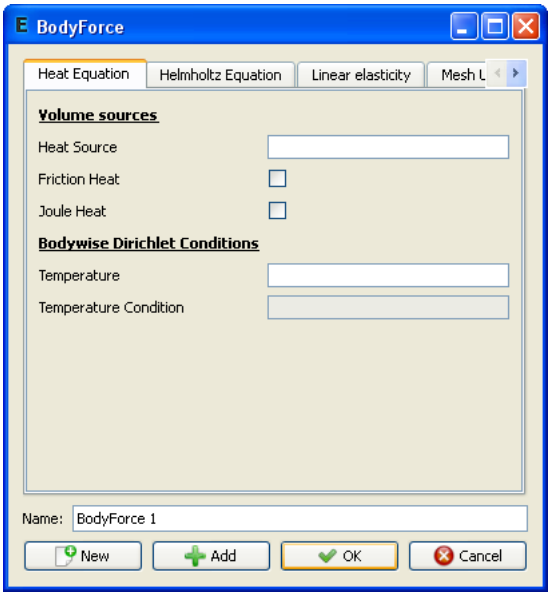

Figure 9: Body force menu.

In this way, also the current mesh and project files get saved in the same directory, avoiding possible inconsistencies later on.

7 Solution and post processing

7.1 Running the solver

Once the solver input file has been generated and the project has been saved, it is possible to actualy solve the problem:

Run → *Start solver*

This will launch either a single process for ElmerSolver (scalar solution) or multiple MPI-processes for ElmerSolver_mpi (parallel solution) depending on the definitions in

Run → *Parallel settings...*

The parallel menu has three group boxes. Usually, the user is supposed to touch only the "General settings" group and select the number of processes to execute. The two remaining groups deal with system commands to launch MPI-processes and external tools for domain decomposition. The parallel menu is greyed out if ElmerSolver_mpi is not present at start-up.

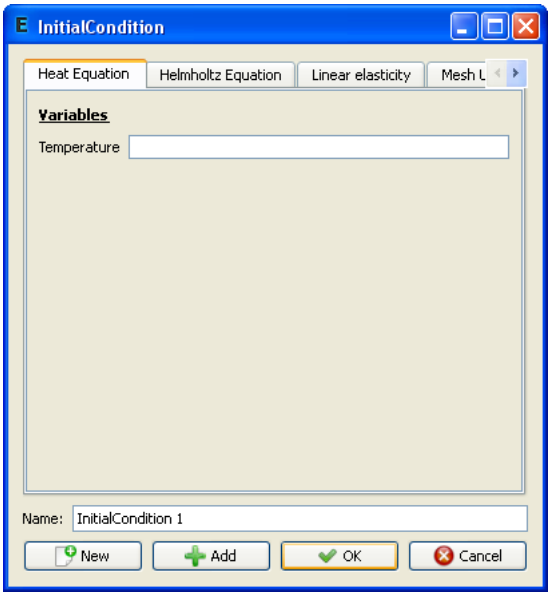

Figure 10: Initial condition menu.

When the solver is running, there is a log window and a convergence monitor from which the iteration may be followed. In case of divergence or other troubles, the solver may be terminated by choosing

Run → *Kill solver*

The solver will finally write a result file for ElmerPost in the project directory. The name of the ep-file is defined in

Model → *Setup...*

7.2 Post preocessor

The post processor is activated from

Run → *Start postprocessor*

This will launch ElmerPost, which will read in the results and displays a contour plot representing the solution. If the results were prodoced by the parallel solver, the domain decomposition used in the calculations will be shown.

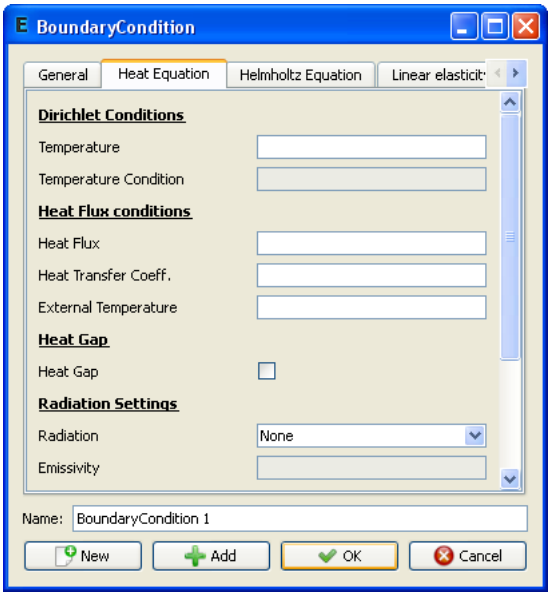

Figure 11: Boundary condition menu.

References

[1] Elmer web pages: http://www.csc.fi/elmer/.

[2] Netgen web pages: http://www.hpfem.jku.at/netgen/.

[3] Qt web pages: http://trolltech.com/products/qt/.

[4] Qwt web pages: http://qwt.sourceforge.net/.

[5] Tetgen web pages: http://tetgen.berlios.de/.

[6] Vtk web pages: http://www.vtk.org/.

A ElmerGUI initialization file

The initialization file for ElmerGUI is located in `ELMERGUI_HOME/edf`. It is called `egini.xml`:

```
<?xml version='1.0' encoding='UTF-8'?>
<!DOCTYPE egini>
<egini version="1.0">

  Show splash screen at startup:
```

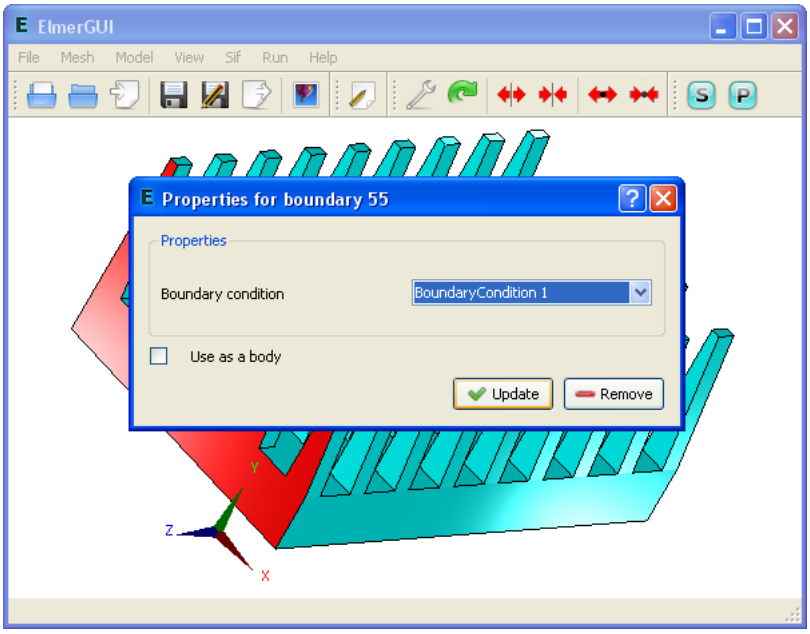

Figure 12: Boundary property editor activated by holding down the ALTGR key while double clicking a surface.

```
<splashscreen> 1 </splashscreen>

Show system tray icon while running:
<systrayicon> 1 </systrayicon>

Show system tray messages:
<systraymessages> 1 </systraymessages>

System tray message duration in millisecons:
<systraymsgduration> 3000 </systraymsgduration>

Check the presence of external components:
<checkexternalcomponents> 0 </checkexternalcomponents>

Hide toolbars:
<hidetoolbars> 0 </hidetoolbars>

Plot convergence view:
<showconvergence> 1 </showconvergence>

Draw background image:
<bgimage> 1 </bgimage>
```

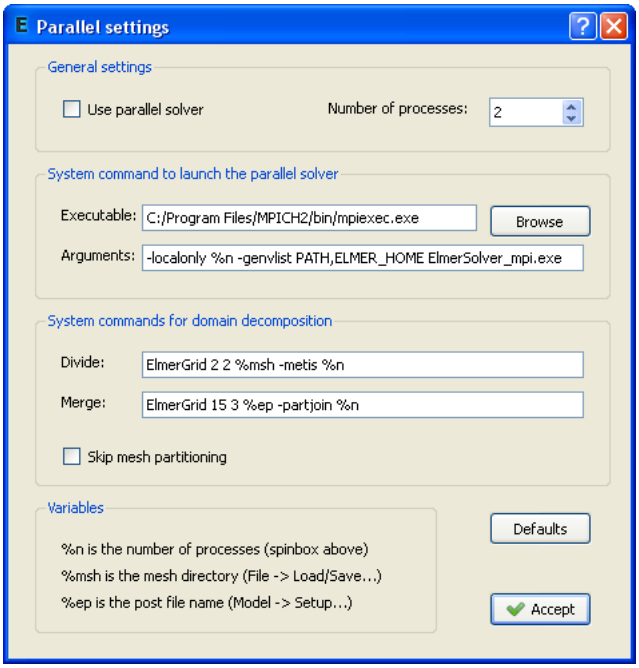

Figure 13: Parallel settings dialog.

```
Background image file:
<bgimagefile> :/images/bgimage.png </bgimagefile>

Align background image to the bottom right corner of the screen:
<bgimagealignright> 0 </bgimagealignright>

Stretch background image to fit the display area (overrides align):
<bgimagestretch> 1 </bgimagestretch>

Maximum number of solvers / equation:
<max_solvers> 10 </max_solvers>

Maximum number of equations:
<max_equations> 10 </max_equations>

Maximum number of materials:
<max_materials> 10 </max_materials>

Maximum number of bodyforces:
<max_bodyforces> 10 </max_bodyforces>

Maximum number of initial conditions:
<max_initialconditions> 10 </max_initialconditions>
```

```
  Maximum number of bodies:
  <max_bodies> 100 </max_bodies>

  Maximum number of bcs:
  <max_bcs> 500 </max_bcs>

  Maximum number of boundaries:
  <max_boundaries> 500 </max_boundaries>

</egini>
```

You may change the default behaviour of ElmerGUI by editing this file. For example, to turn off the splash screen at start up, change the value of the tag `<splshscreen>` from 1 to 0. To change the background image, enter a picture file name in the `<bgimagefile>` tag. You might also want to increase the default values for solvers, equations, etc., in case of very complex models. This will slightly increase the memory usage of ElmerGUI.

B ElmerGUI definition files

The directory `ELMERGUI_HOME` contains a subdirectory called "edf". This is the place where all ElmerGUI definition files (ed-files) reside. The definition files are XML-formatted text files which define the contents and appearance of the Model menu.

The ed-files are loaded iteratively from the edf-directory once and for all when ElmerGUI starts. Later, it is possible to view and edit their contents by choosing

File → Definitions...

An ed-file has the following structure:

```
<?xml version='1.0' encoding='UTF-8'?>
<!DOCTYPE edf>
<edf version="1.0">
   [PDE block]
   [PDE block]
   ...
   [PDE block]
</edf>
```

The structure of a [PDE block] is the following:

```
<PDE Name="My equation">
   <Name>
      My equation
```

```
   </Name>
   ...
   <Equation>
      [Widget block]
   </Equation>
   ...
   <Material>
      [Widget block]
   </Material>
   ...
   <BodyForce>
      [Widget block]
   <BodyForce>
   ...
   <InitialCondition>
      [Widget block]
   </InitialCondition>
   ...
   <BoundaryCondition>
      [Widget block]
   </BoundaryCondition>
</PDE>
```

Note that the name of the PDE is defined redundantly in two occurances.

The basic stucture of a [Widget block] is the following:

```
<Parameter Widget="Label">
   <Name> My label </Name>
</Parameter>
...
<Parameter Widget="Edit">
   <Name> My edit box </Name>
   <Type> Integer </Type>
   <Whatis> Meaning of my edit box </Whatis>
</Parameter>
...
<Parameter Widget="CheckBox">
   <Name> My check box </Name>
   <Type> Logical </Type>
   <Whatis> Meaning of my check box </Whatis>
</Parameter>
...
<Parameter Widget="Combo">
   <Name> My combo box </Name>
   <Type> String </Type>
   <Item> <Name> My 1st item </Name> </Item>
   <Item> <Name> My 2nd item </Name> </Item>
   <Item> <Name> My 3rd item </Name> </Item>
   <Whatis> Meaning of my combo box </Whatis>
</Parameter>
```

There are four types of widgets available:

- Label (informative text)
- CheckBox (switches)
- ComboBox (selection from list)
- LineEdit (generic variables)

Each widget must be given a name and and a variable type: logical, integer, real, or string. It is also a good practice to equip the widgets with tooltips explaining their purpose and meaning as clearly as possible.

Below is a working example of a minimal ElmerGUI definition file. It will add "My equation" to the equation tabs in the Model menu, see Figure N. The file is called "sample.edf" and it should be placed in `ELMERGUI_HOME/edf`.

```
<?xml version='1.0' encoding='UTF-8'?>
<!DOCTYPE edf>
<edf version="1.0">
  <PDE Name="My equation">
    <Name> My equation </Name>
    <Equation>
      <Parameter Widget="Label">
        <Name> My label </Name>
      </Parameter>
      <Parameter Widget="Edit">
        <Name> My edit box </Name>
        <Type> Integer </Type>
        <Whatis> Meaning of my edit box </Whatis>
      </Parameter>
      <Parameter Widget="CheckBox">
        <Name> My check box </Name>
        <Type> Logical </Type>
        <Whatis> Meaning of my check box </Whatis>
      </Parameter>
      <Parameter Widget="Combo">
        <Name> My combo box </Name>
        <Type> String </Type>
        <Item> <Name> My 1st item </Name> </Item>
        <Item> <Name> My 2nd item </Name> </Item>
        <Item> <Name> My 3rd item </Name> </Item>
        <Whatis> Meaning of my combo box </Whatis>
      </Parameter>
    </Equation>
  </PDE>
</edf>
```

More sophisticated examples with different tags and attribites can be found from the XML-files in `ELMERGUI_HOME/edf`.

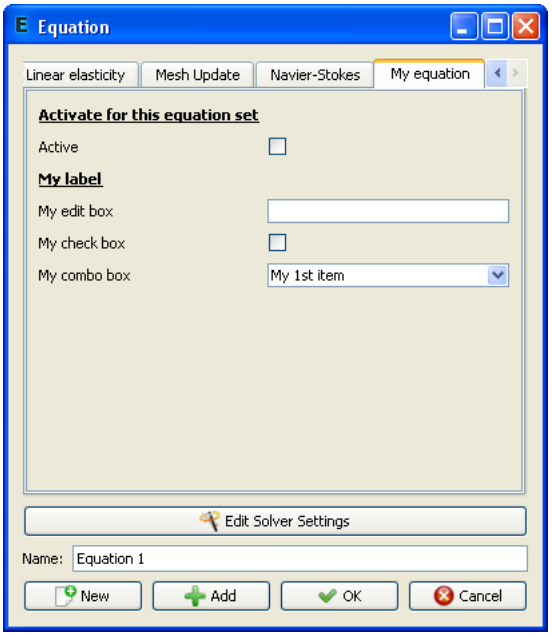

Figure 14: Equation tab in Model menu produced by the sample ed-file.

C Elmer mesh files

mesh.header

```
nodes elements boundary-elements
types
type1 elements1
type2 elements2
...
typeN elementsN
```

mesh.nodes

```
node1 tag1 x1 y1 z1
node2 tag2 x2 y2 z2
...
nodeN tagN xN yN zN
```

mesh.elements

```
element1 body1 type1 n11 ... n1M
element2 body2 type2 n21 ... n2M
...
elementN bodyN typeN nN1 ... nNM
```

mesh.boundary

```
element1 boundary1 parent11 parent12 n11 ... n1M
element2 boundary2 parent21 parent22 n21 ... n2M
...
elementN boundaryN parentN1 parentN2 nN1 ... nNM
```