Elmer Basic Course

Mikko Lyly
Thomas Zwinger
Peter Råback
CSC – Scientific Computing Ltd.
21. - 22. October 2009

21. October

- Welcome
- Introduction to CSC's services
- Introduction to Elmer
- Lunch
- Introduction continued
- Hands on tutorials

22. October

- Tutorials continued
- Lunch
- Programming with Elmer
- Special cases

Introduction to Elmer

- Open source FEM program
- Main emphasis on multphysics (combinations of physical models, PDE systems)
- Scalability (system size & number of procs)
- Powerful numerical methods
- Relatively easy to extend
- Does not provide CAD tools

Introduction to Elmer

- Elmer is intended for...
  - Research: Implementation of new models
  - Parallel computing: Very large models
  - Prototyping: Variation of parameters etc.
  - Education: FEM courses in universities

- Elmer is NOT intended for...
  - Complex CAD modeling
  - Automatically discretizing everything

Introduction to Elmer

- Main components:
- Preprocessor: ElmerGUI
- Solution engine: ElmerSolver
- Postprocessor: ElmerPost (ElmerGUI)
- Mesh generation & manipulaion: ElmerGrid

Installation

- Windows: Binaries available from nic.funet.fi. Needs Visual C++ 2008 Redistributable package installed. User account should have admin rights.
- Linux: Source code available from SF.net. Needs a fully functional build environment (compilers, libraries, utilities).
- Mac: Binaries provided by TrueFlaw Ltd.

Resources

- Official site: http://www.csc.fi/elmer
- Wiki and BB: http://elmerfem.org
- SF: http://sourceforge.net/projects/elmerfem/
- Documentation: www.csc.fi/english/pages/elmer/documentation
- Nightly builds: http://www.nic.funet.fi/pub/sci/physics/elmer/

Getting started: Small example

- Typical analysis: Heat conduction in solid
- Geometry modeling e.g. with Salome
- CAD import in ElmerGUI
- Meshing with ElmerGUI
- Analysis with ElmerSolver
- Visualization with ElmerPost

Getting started: continued

- Work flow controlled by ElmerGUI...
- ElmerGUI prepares input files for ElmerSolver.
- ElmerSolver prepares input files for ElmerPost.
- ElmerGUI coordinates the work flow.
- Once the project is saved, it is possible to (re)run the project from command line:

```
$ cd SmallSample
$ ElmerSolver
```

Getting started: continued

- Basic file structure produced by ElmerGUI:

```
SmallSample | ELMERSOLVER_STARTINFO
            | case.sif
            | egproject.xml
            | cube.stp
            | mesh.header
            | mesh.nodes
            | mesh.elements
            | mesh.boundary
```

Getting started: continued

- `model.stp` is the original CAD file
- `case.sif` is the input file for ElmerSolver
- `ELMERSOLVER_STARTINFO` is the startup file
- `egproject.xml` is the state of the GUI
- `mesh.*` are the finite element mesh files
- Usual modeling strategy is to create the project files by ElmerGUI (or by hand), and to continue on command line.

Solver Input Files

- Main main ingredients is the SIF
- SIF has a block structure:

```
Header
      …
End

Material
    …
End
```

Solver Input Files

- Relevant blocks (N represents an integer):

```
Header
Simulation
Constants
Body N
Solver N
Equation N
Material N
Body Force N
Boundary Condition N
Initial Condition N
```

Solver Input Files: Header

- Header-block defines the work directories (ElmerGUI: Model → Setup...):

```
Header
  CHECK KEYWORDS Warn
  Mesh DB "." "."
  Include Path ""
  Results Directory ""
End
```

Solver Input Files: Simulation

- Simulation-block defines verbosity, coordinate systems, output files, and time marching methods (ElmerGUI: Model → Setup...):

```
Simulation
   Max Output Level = 4
   Coordinate System = Cartesian
   Coordinate Mapping(3) = 1 2 3
   …
   Post File = case.ep
End
```

Solver Input Files: Constants

- Constants-block defines generic constants (ElmerGUI: Model → Setup...):

```
Constants
  Gravity(4) = 0 -1 0 9.82
  Stefan Boltzmann = 5.67e-08
  Permittivity of Vacuum = 8.8542e-12
  Boltzmann Constant = 1.3807e-23
  Unit Charge = 1.602e-19
End
```

Solver Input Files: Body

- Body-block assigns materials, loads, equations, etc., to computational domains (generation handled internally by ElmerGUI):

```
Body 1
   Target bodies(3) = 1 2 5
   Name = "Body 1"
   Equation = 1
   Material = 1
   Body Force = 1
End
```

Solver Input Files: Solver

- Solver-block defines the numerical methods and parameters related to an individual PDE (ElmerGUI: Model → Equation → Add):

```
Solver 1
   Equation = Heat equation
   Procedure = "HeatSolve" "HeatSolver"
   Linear system solver = Iterative
   Linear system iterative method = CG
   ...
End
```

Solver Input Files: Equation

- Equation-block defines the final PDE system, i.e., Solvers to execute (handled internally in ElmerGUI):

```
Equation 1
   Name = “My system”
   Active Solvers(3) = 1 2 3
End
```

Solver Input Files: Material

- Material-block defines problem dependent parameters (ElmerGUI: Model → Material → Add...):

```
Material 4
   Heat Conductivity = 237.0
   Heat Capacity = 897.0
   Density = 2700.0
   Youngs Modulus = 70e9
   ...
End
```

Solver Input Files: BCs

- Bondary Condition-block defines the BCs (ElmerGUI: Model → Boundary condition → Add):

```
Boundary Condition 7
   Target Boundaries(5) = 1 4 8 10 34
   Temperature = 300
   ...
End
```

Solver Input Files: Initial Cond.

- Initial Condition-block defines the state for t=0 (ElmerGUI: Model → Initial condition → Add...):

```
Initial Condition 1
   Target bodies(1) = 1
   Temperature = 300
   ...
End
```

ELMERSOLVER_STARTINFO

- The start info file defines the name of the SIF and the number of processors to use (handled internally in ElmerGUI). The file has two lines:

```
case.sif
1
```

Mesh Files: Nodes

- Nodes of the FE-mesh are listed in a file called mesh.nodes. The format is the following:

```
N(1) Tag(1) X(1) Y(1) Z(1)
N(2) Tag(2) X(2) Y(2) Z(2)
…
N(n) Tag(n) X(n) Y(n) Z(n)
```

Mesh Files: Elements

- Elements are listed in mesh.elements (here [] denotes a list of integers):

```
N(1) Body(1) Code(1) [Nodes(1)]
N(2) Body(2) Code(2) [Nodes(2)]
…
N(n) Body(n) Code(n) [Nodes(n)]
```

Mesh Files: Boundaries

- Boundaries are listed in mesh.boundary (Left and Right are the so called parent elements):

```
N(1) BC(1) Left(1) Right(1) Code(1) [Nodes(1)]
N(2) BC(2) Left(2) Right(2) Code(2) [Nodes(2)]
…
N(n) BC(n) Left(n) Right(n) Code(n) [Nodes(n)]
```

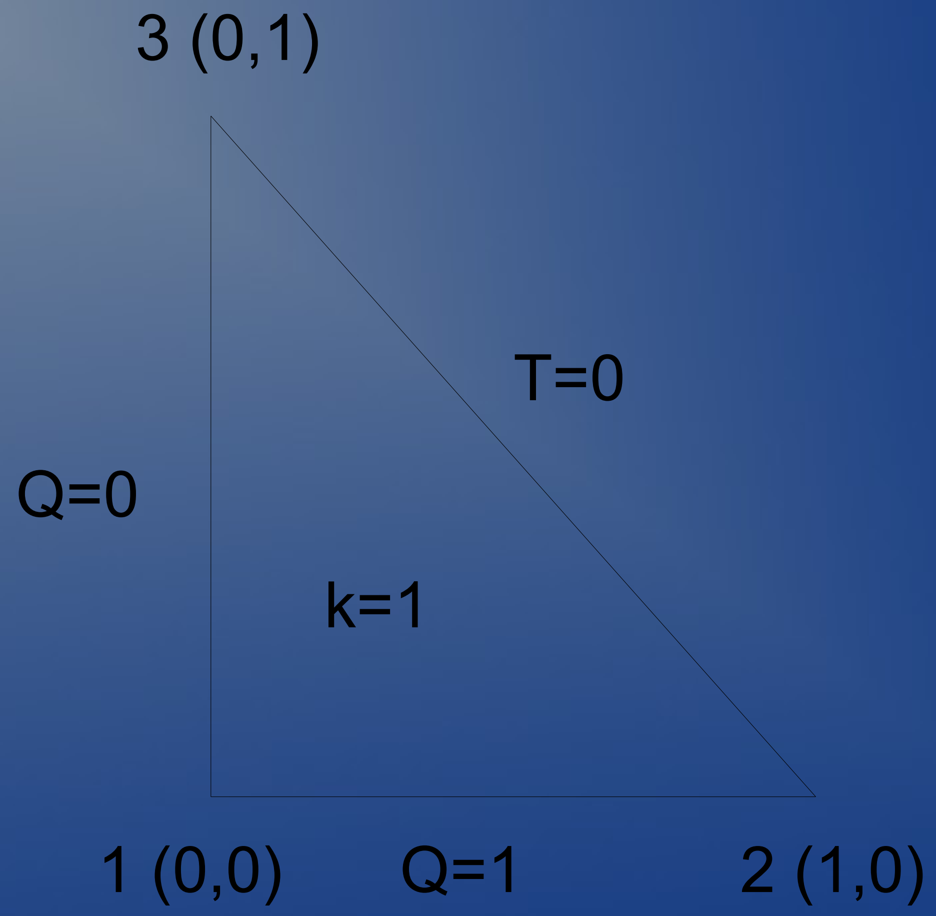
Mesh Files: Header

- Finally, there is mesh.header (i.e. summary):

```
NofNodes NofElements NofBoundaryElements
NofElementTypes
Type(1) NofElements(1)
Type(2) NofElements(2)
…
Type(n) NofElements(n)
```

CLI: Minimal Example

- Heat conduction in triangular domain with one element:



CLI: Minimal Example

- mesh.nodes:

```
1 -1 0.0 0.0 0.0
2 -1 1.0 0.0 0.0
3 -1 0.0 1.0 0.0
```

- mesh.boundary:

```
1 1 1 0 202 1 2
2 2 1 0 202 2 3
3 3 1 0 202 3 1
```

CLI: Minimal Example

- mesh.elements:

```
1 1 303 1 2 3
```

- mesh.header:

```
3 1 3
2
202 3
303 1
```

CLI: Minimal Example

- case.sif:

```
Header
  CHECK KEYWORDS Warn
  Mesh DB "." "."
End

Simulation
  Simulation Type = Steady State
  Post File = case.ep
End
```

CLI: Minimal Example

```
Body 1
  Target Bodies(1) = 1
  Equation = 1
  Material = 1
End

Solver 1
  Equation = Heat Equation
  Linear System Solver = Direct
End
```

CLI: Minimal Example

```
Equation 1
  Active Solvers(1) = 1
End

Material 1
  Heat Conductivity = 1
  Density = 1
End
```

CLI: Minimal Example

```
Boundary Condition 1
  Target Boundaries(1) = 2
  Temperature = 0
End

Boundary Condition 2
  Target Boundaries(1) = 3
  Heat Flux = 1
End
```

CLI: Minimal Example

- ELMERSOLVER_STARTINFO:

```
case.sif
1
```

- Solve the problem from CLI by typing:

```
$ ElmerSolver
```

- Launch the post processor:

```
$ ElmerPost "readfile case.ep"
```