Post-processing utilities within ElmerSolver

ElmerTeam
CSC – IT Center for Science, Finland

CSC, 2018

Postprocessing utilities in ElmerSolver

- Saving data
  - FEM data
  - Line data
  - Scalars data
  - Grid data
- Computing data
  - Derived fields (gradient, curl, divecgence,…)
  - Data reduction & filtering
  - Creating fields of material properties
- The functionality is usually achieved by use of atomic auxialiry solvers

Computing derived fields

- Many solvers have internal options or dedicated post-processing solvers for computing derived fields
  - E.g. stress fields by the elasticity solvers
  - E.g. **MagnetoDynamicsCalcFields**
- Elmer offers several auxiliary solvers that may be used in a more generic way
  - **`SaveMaterials`**: makes a material parameter into field variable
  - **`StreamlineSolver`**: computes the streamlines of 2D flow
  - **`FluxSolver`**: given potential, computes the flux $q = - c \nabla \phi$
  - **`VorticitySolver`**: computes the vorticity of flow, $w = \nabla \times \phi$
  - **`PotentialSolver`**: given flux, compute the potential $- c \nabla \phi = q$
  - **`FilterTimeSeries`**: compute filtered data from time series (mean, fourier coefficients,...)
  - ...

Derived nodal data

- By default Elmer operates on distributed fields but sometimes nodal values are of interest
  - Multiphysics coupling may also be performed alternatively using nodal values for computing and setting loads
- Elmer computes the nodal loads from $Ax-b$ where A, and b are saved before boundary conditions are applied
  - **`Calculate Loads = True`**
- This is the most consistant way of obtaining boundary loads
- Note: the nodal data is really pointwise
  - expressed in units N, C, W etc.
    (rather than N/m^2, C/m^2, W/m^2 etc.)
  - For comparison with distributed data divided by the ~size of the surface elements

Derived lower dimensional data

- Derived boundary data
  - SaveLine: Computes fluxes on-the-fly
- Derived lumped (or oD) data
  - SaveScalars: Computes a large number of different quantities on-the-fly
  - FluidicForce: compute the fluidic force acting on a surface
  - ElectricForce: compute the electrostatic froce using the Maxwell stress tensor
  - Many solvers compute lumped quantities internally for later use (Capacitance, Lumped spring,…)

Exporting FEM data: ResultOutputSolve

- Currently recommened format is **VTU**
  - XML based unstructured VTK
  - Has the most complete set of features
  - Old ElmerPost format (with suffix .ep) is becoming obsolite
  - Simple way to save VTU files: **`Post File = file.vtu`**
- ResultOutputSolve offers additionally several formats
  - vtk: Visualization tookit legacy format
  - vtu: Visualization tookit XML format
  - Gid: GiD software from CIMNE: http://gid.cimne.upc.es
  - Gmsh: Gmsh software: http://www.geuz.org/gmsh
  - Dx: OpenDx software

Exporting 2D/3D data: ResultOutputSolve

An example shows how to save data in unstructured XML VTK (.vtu) files to directory ”results” in single precision binary format.

```
Solver n
  Exec Solver = after timestep
  Equation = "result output"
  Procedure = "ResultOutputSolve""ResultOutputSolver"
  Output File Name = "case"
  Output Format = String ”vtu”
  Binary Output = True
  Single Precision = True
End
```

Saving 1D data: SaveLine

- Lines of interest may be defined on-the-fly
- Data can either be saved in uniform 1D grid,
  or where element faces and lines intersect
- Flux computation using integration points on the boundary –
  not the most accurate
- By default saves all existing field variables

Saving 1D data: SaveLine...

```
Solver n
 Equation = "SaveLine"
 Procedure = File "SaveData" "SaveLine"
 Filename =  "g.dat"
 File Append = Logical True
 Polyline Coordinates(2,2) = Real 0.0 1.0 0.0 2.0
End

Boundary Condition m
  Save Line = Logical True
End
```

Computing and saving oD data: SaveScalars

Operators on bodies

- Statistical operators
  - Min, max, min abs, max abs, mean, variance, deviation, rms
- Integral operators (quadratures on bodies)
  - volume, int mean, int variance, int rms
  - Diffusive energy, convective energy, potential energy

Operators on boundaries

- Statistical operators
  - Boundary min, boundary max, boundary min abs, max abs, mean, boundary variance, boundary deviation, boundary sum, boundary rms
  - Min, max, minabs, maxabs, mean
- Integral operators (quadratures on boundary)
  - area
  - Diffusive flux, convective flux

Other operators

- nonlinear change, steady state change, time, timestep size,...

Saving oD data: SaveScalars...

```
Solver n
  Exec Solver = after timestep
  Equation = String SaveScalars
  Procedure = File "SaveData" "SaveScalars"
  Filename = File "f.dat"
  Variable 1 = String Temperature
  Operator 1 = String max
  Variable 2 = String Temperature
  Operator 2 = String min
  Variable 3 = String Temperature
  Operator 3 = String mean
End

Boundary Condition m
  Save Scalars = Logical True
End
```

Slots for executing postprocessing solvers

- Often the postprocessing solver need to computed only at desired slots, not at every time-step or coupled system iteration
- The execution is controlled by the ”Exec Solver” keyword
  - Exec Solver = before simulation
  - Exec Solver = after simulation
  - Exec Solver = before timesteo
  - Exec Solver = after timestep
  - Exec Solver = before saving
  - Exec Solver = after saving
- The before/after saving slot is controlled by the output intervals
  - Derived solvers often use the ”before saving” slot
  - Data is often saved with the ”after saving” slot

Case: TwelveSolvers

Natural convection with ten auxialiary solvers

Case: Motivation

- The purpose of the example is to show the flexibility of the modular structure
- The users should not be afraid to add new atomistic solvers to perform specific tasks
- A case of 12 solvers is rather rare, yet not totally unrealitistic

Case: preliminaries

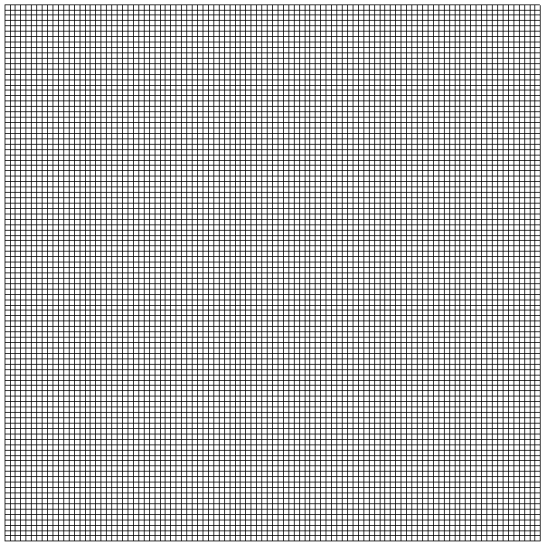
- Square with hot wall on right and cold wall on left
- Filled with viscous fluid
- Bouyancy modeled with Boussinesq approximation
- Temperature difference initiates a convection roll

Case: 12 solvers

1. **HeatSolver**
2. **FlowSolver**

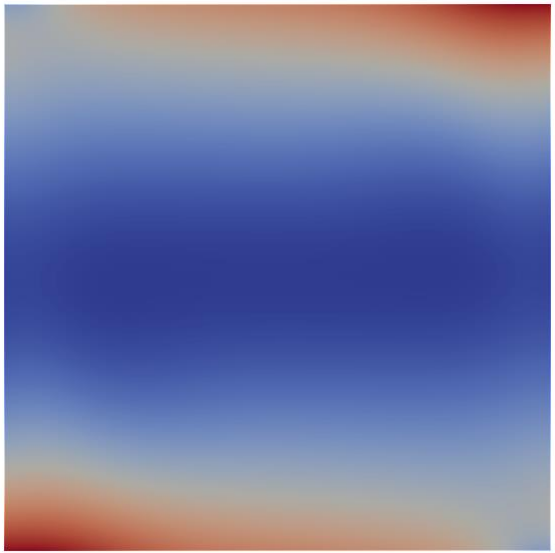
---

3. **FluxSolver**: solve the heat flux
4. **StreamSolver**: solve the stream function
5. **VorticitySolver**: solve the vorticity field (curl of vector field)
6. **DivergenceSolver**: solve the divergence
7. **ShearrateSolver**: calculate the shearrate
8. **IsosurfaceSolver**: generate an isosurface at given value
9. **ResultOutputSolver**: write data
10. **SaveGridData**: save data on uniform grid
11. **SaveLine**: save data on given lines
12. **SaveScalars**: save various reductions

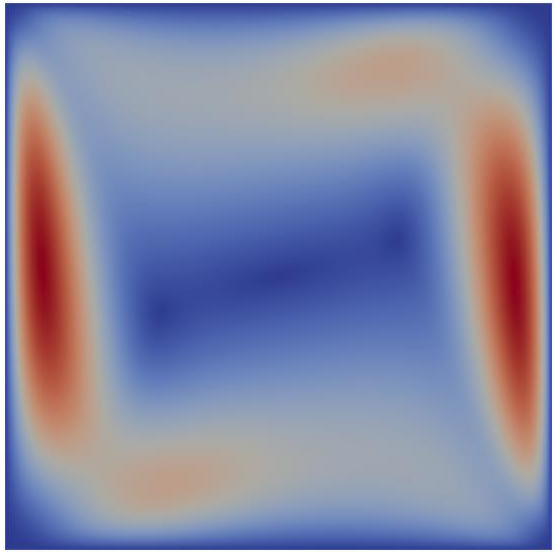


Mesh of 10000 bilinear elements

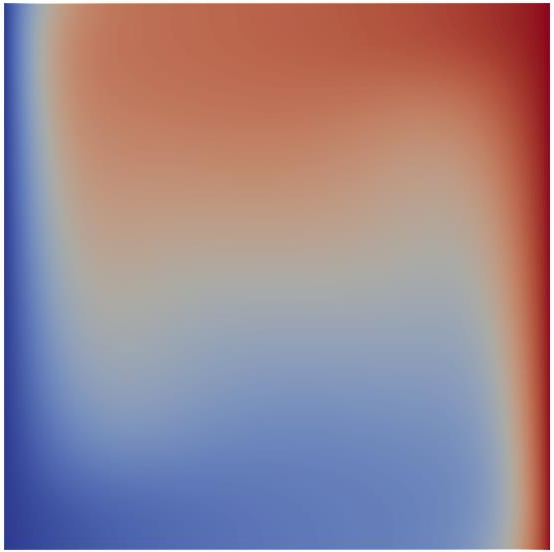
Primary fields for natural convection



Pressure

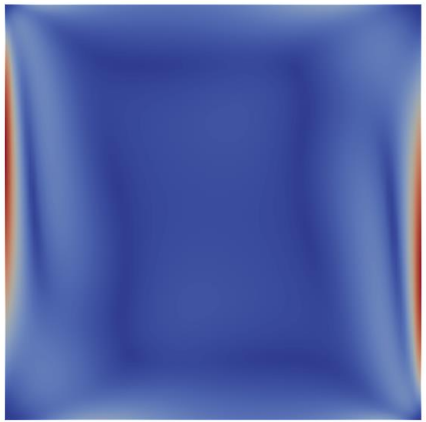


Velocity

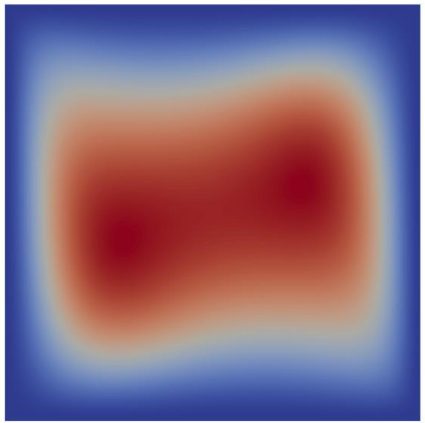


Temperature

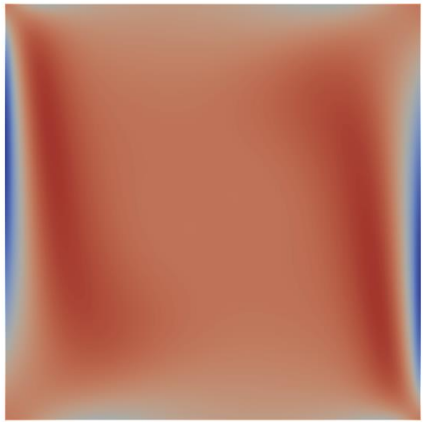
Derived fields for Navier-Stokes solution



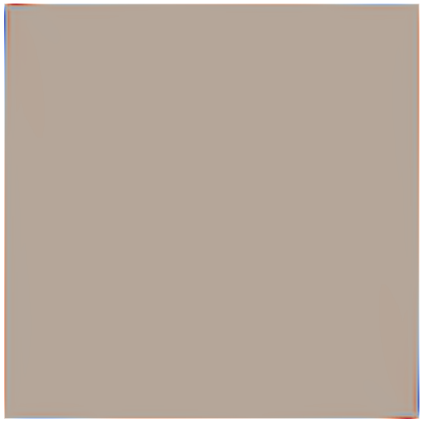
Shearrate field



Stream function

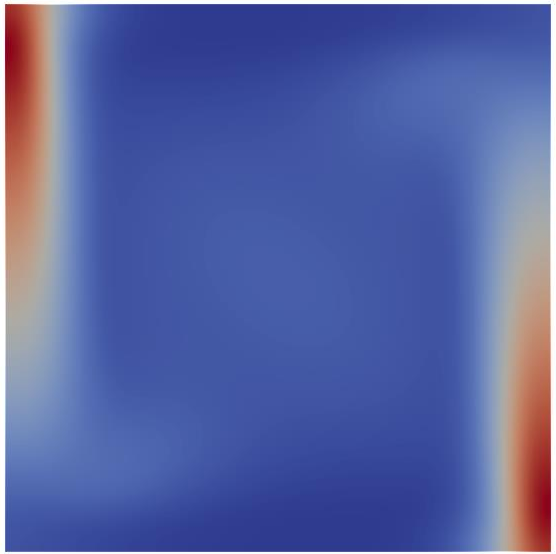


Vorticity field

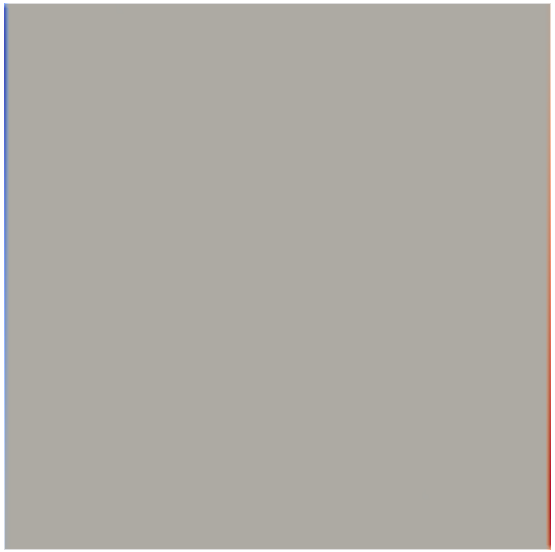


Divergence field

Derived fields for heat equation



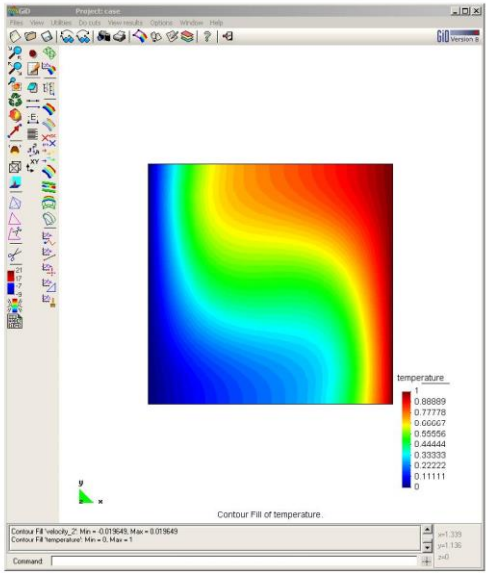
Heat flux



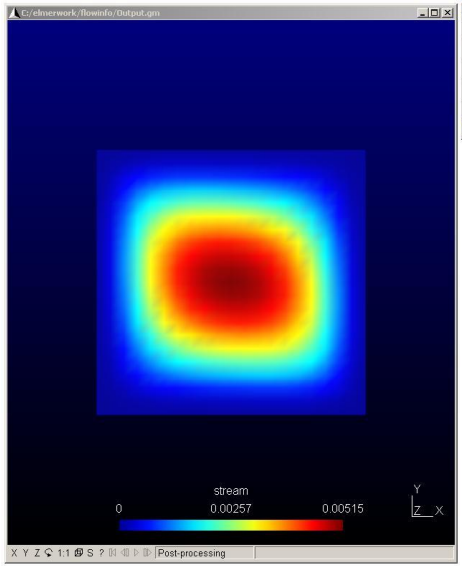
Nodal heat loads

- Nodal loads only occur at boundaries (nonzero heat source)
- Nodal loads are associated to continuous heat flux by element size factor

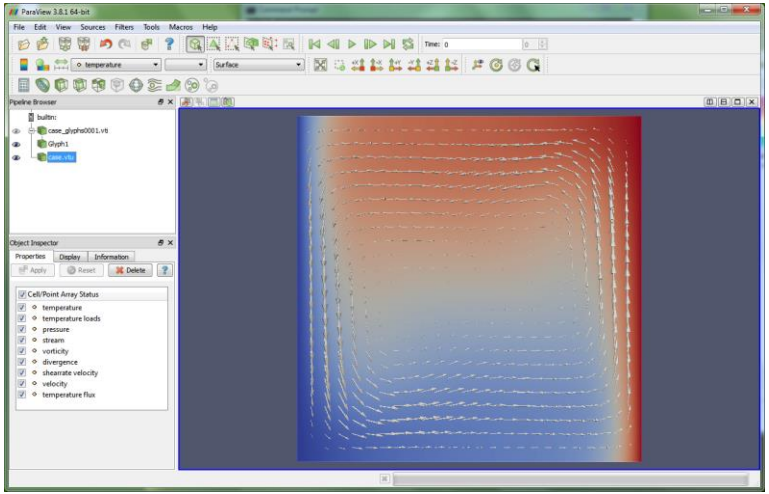
Visualization in differen postprocessors



GiD



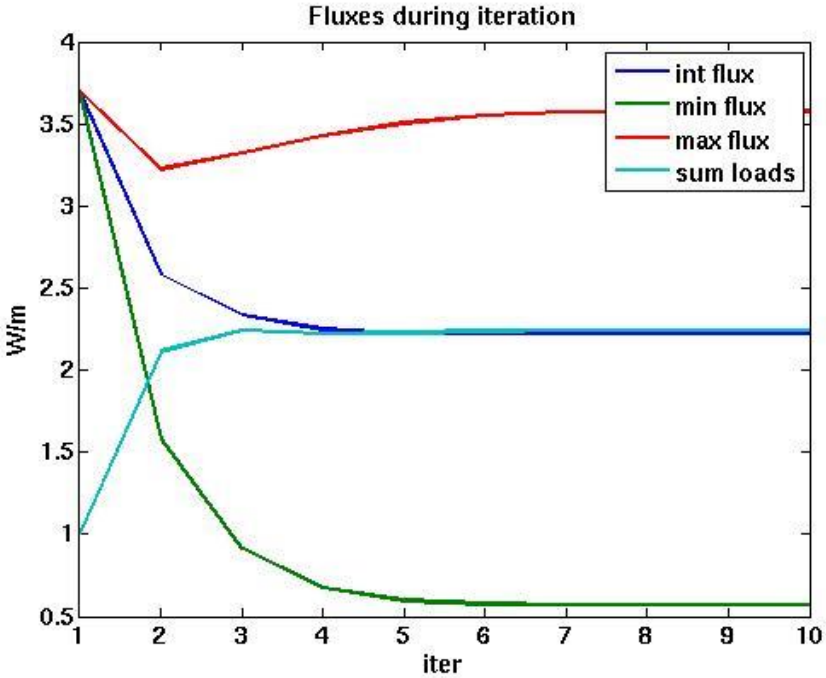
gmsh



Paraview

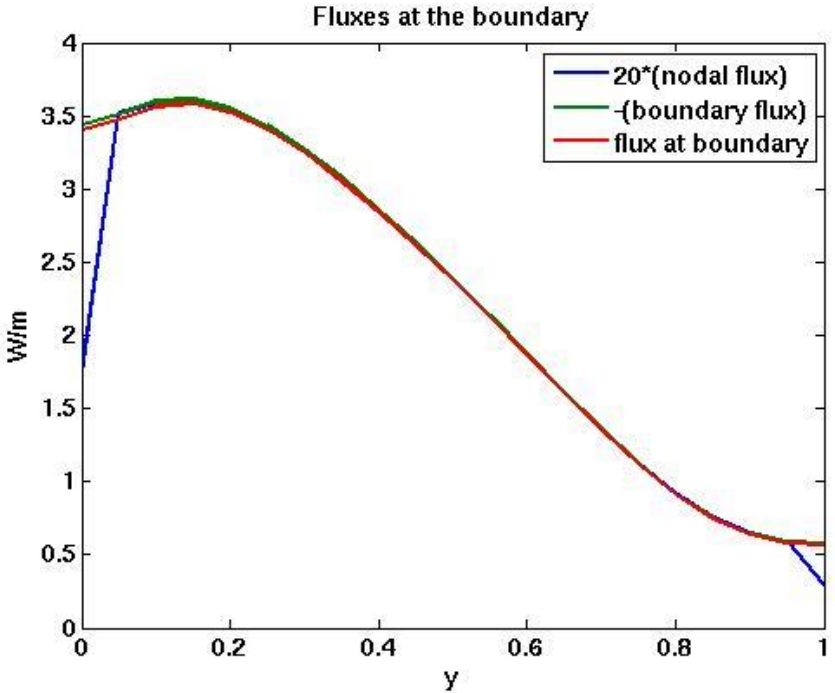
Example: total flux

- Saved by SaveScalars
- Two ways of computing the total flux give different approximations
- When convergence is reached the agreement is good



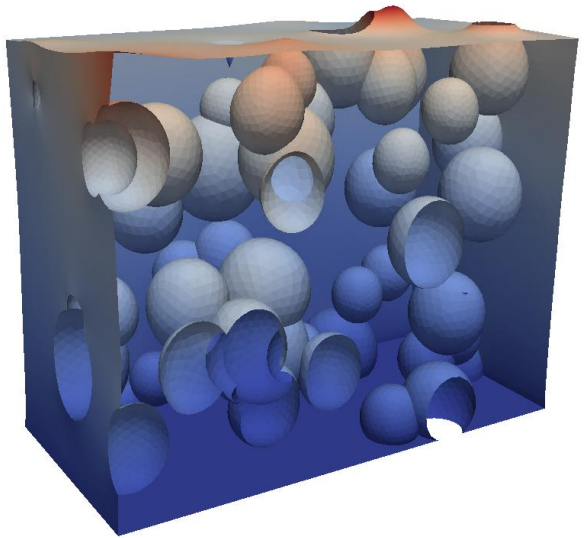
Example: boundary flux

- Saved by SaveLine
- Three ways of computing the boundary flux give different approximations
- At the corner the nodal flux should be normalized using only *h/2*



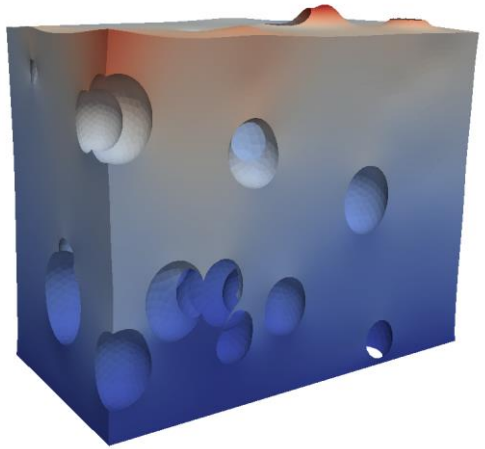
Example, saving boundaries in .sif file

```
Solver 2
  Exec Solver = Always
  Equation = "result output"
  Procedure = "ResultOutputSolve”
"ResultOutputSolver"
  Output File Name = case
  Vtu Format = Logical True
  Save Boundaries Only = Logical True
End
```



Example, File size in Swiss Cheese

- Memory consumption of vtu-files (for Paraview) was studied in the "swiss cheese" case
- The ResultOutputSolver with different flags was used to write output in parallel
- Saving just boundaries in single precision binary format may save over 90% in files size compared to full data in ascii
- With larger problem sizes the benefits are amplified



Binary output	Single Prec.	Only bound.	Bytes/node
-	X	-	376.0
X	-	-	236.5
X	X	-	184.5
X	-	X	67.2
X	X	X	38.5

Manually editing the command files

- Only the most important solvers and features are supported by the GUI
- Minor modifications are most easily done by manual manipulation of the files
- The tutorials, test cases and documentation all include usable sif file pieces
- Use your favorite text editor (emacs, notepad++,...) and copy-paste new definitions to your .sif file
- If your additiones were sensible you can rerun your case
- Note: you cannot read in the changes made in the .sif file

Exercise

- Study the command file with 12 solvers
- Copy-paste an appropriate solver from there to some existing case of your own
  - ResultOutputSolver for VTU output
  - StreamSolver, VorticitySolver, FluxSolver,…
- Note: Make sure that the numbering of Solvers is consistant
  - Solvers that involve finite element solution you need to activate by `Active Solvers`
- Run the modified case
- Visualize results in Paraview in different ways

Using tests as a starting point

- There are over 500 consistancy tests that come with the Elmer distribution
  - The hope is to minimize the propability of new bugs
- The tests are small for speedy computation
- Step-by-step instructions
  1. Go to tests at $ELMER_HOME/tests
  2. Choose a test case relevant to you (by name, or by grep)
     - Look in Models manual for good search strings
  3. Copy the tests to your working directory
  4. Edit the sif file
     - Activate the output writing: Post File
     - Make the solver more verbose: Max Output Level
  5. Run the case (see runtest.cmake for the meshing procedure)
     - Often just: ElmerSolver
  6. Open the result file to see what you got
  7. Modify the case and rerun etc.

Conclusions

- It is good to think in advance what kind of data you need
  - 3D volume and 2D surface data
  - Derived fields
  - 1D line data
  - 0D lumped data
- Internal strategies may allow better accuracy than doing the analysis with external postprocessing software
  - Consistent use of basis functions to evaluate the data
- Often the same reduction operations may be done also at later stages but with significantly greater effort