Postprocessing Utilities of Elmer software

D.Sc. Peter Råback

CSC, Finnish IT Center for Science

Outline

- **Derived fields**
- **Derived nodal data**
- **Derived lower dimensional data**
- **Saving 0D data**
- **Saving 1D data**
- **Saving distributed data**
- **Example**

Derived fields

- **Often it is desirable to compute a derived field from the solution**
- **Elmer offers several small auxiliary routines for this purpose**
  - Many solvers have internal options for computing derived fields (fluxes, heating powers,…)
  - SaveMaterials: makes a material parameter into field variable
  - Streamlines: computes the streamlines of 2D flow
  - FluxComputation: 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 $- c \nabla \phi = q$
  - Filtered Data: compute filtered data from time series (mean, fourier coefficients,…)
  - …
- **Usually auxiliary data need to be computed only after the iterative solution is ready**
  - Exec Solver = after timestep
  - Exec Solver = after all

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**
- **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 elements

Derived lower dimensional data

- **Derived boundary data**
  - SaveLine: Computes fluxes on-the-fly
- **Derived lumped (or 0D) 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,…)

Saving 0D 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
```

Saving 0D data: SaveScalars

Operators on bodies

- **Statistical operators**
  - Min, max, min abs, max abs, mean, variance, deviation
- **Integral operators (quadratures on bodies)**
  - volume, int mean, int variance
  - 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
  - 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 1D data: SaveLine…

```
Solver n
 Equation = "SaveLine"
 Procedure = File "SaveData" "SaveLine"
 Filename =  "g.dat"
 File Append = Logical True
 Polyline Coordinates(2,2) = Real 0.25 -1 0.25 2.0
End

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

Saving 1D data: SaveLine

- **Lines of interest may be defined on-the-fly**
- **Flux computation using integration points on the boundary**
- **By default saves all existing field variables**

Exporting distributed data: ResultOutputSolve

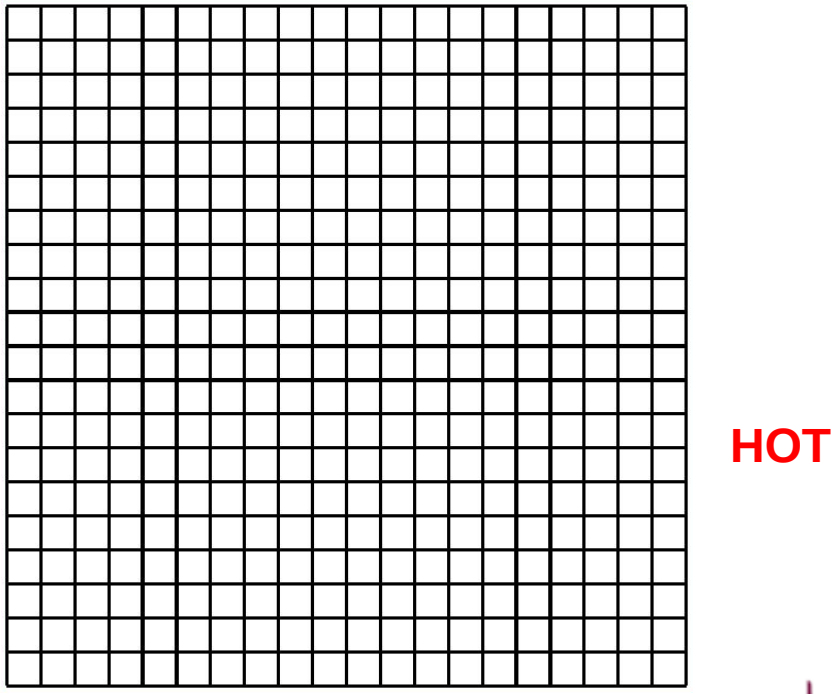
```
Solver n
  Exec Solver = after all
  Equation = "result output"
  Procedure = "ResultOutputSolve" "ResultOutputSolver"
  Output File Name = "case"
  Output Format = String ”vtu”
  Scalar Field 1 = String Temperature
  Scalar Field 2 = String Pressure
  Vector Field 1 = String Velocity
End
```

Exporting distributed data: ResultOutputSolve

- **Apart from saving the results in .ep format it is possble to use other postrprocessing tools**
- **ResultOutputSolve offers 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
- **vtu offers parallel data handling capabilities and may therefore be desirable in large computations**
- **binary formats missing**
  - Output files often quite large

Example: preliminaries

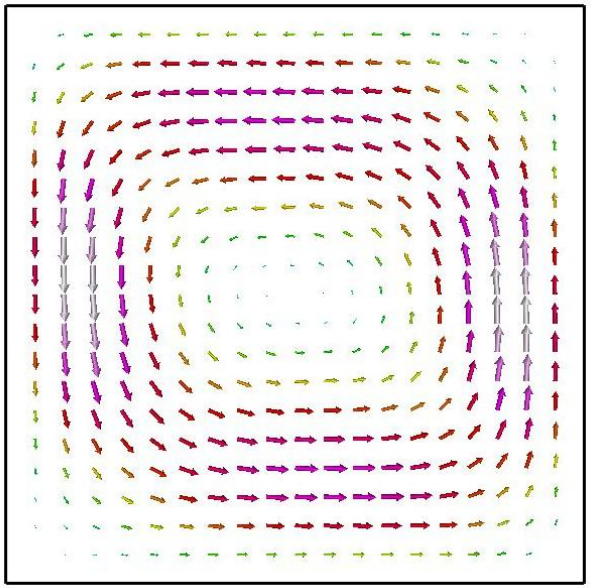
- **Square with hot wall on right and cold wall on left**
- **Filled with viscous fluid**
- **Bouyancy modeled with Boussinesq approximation**
- **Temperature difference initiates a convection roll**

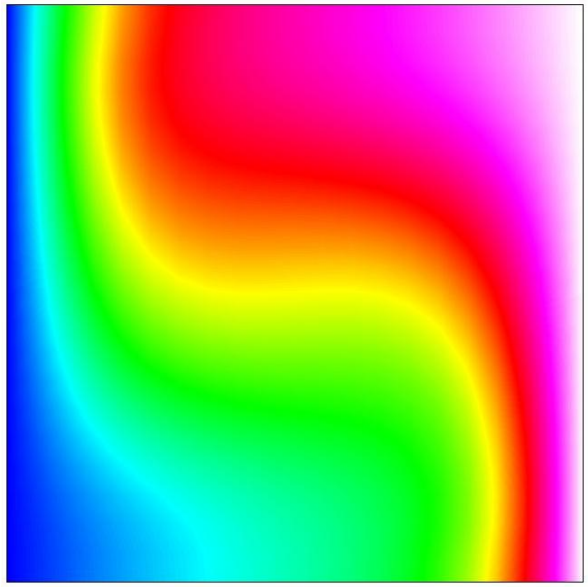


Case: solvers

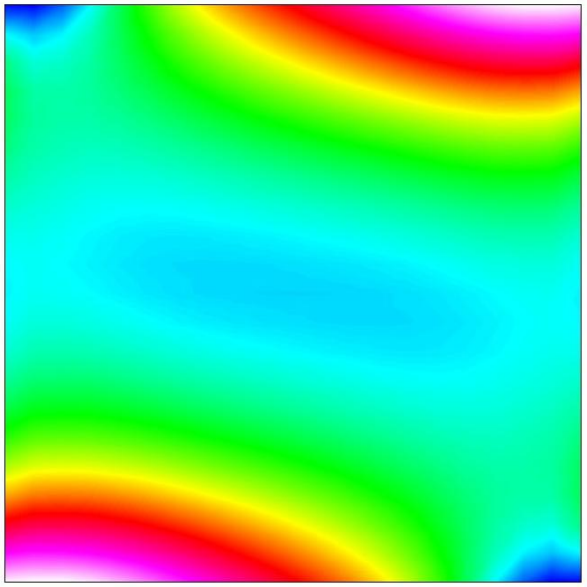
1. **Heat Equation**
2. **Navier-Stokes**
3. **FluxSolver**
4. **StreamSolver**
5. **VorticitySolver**
6. **ResultOutputSolver**
7. **SaveLine**
8. **SaveScalars**

Example: solution

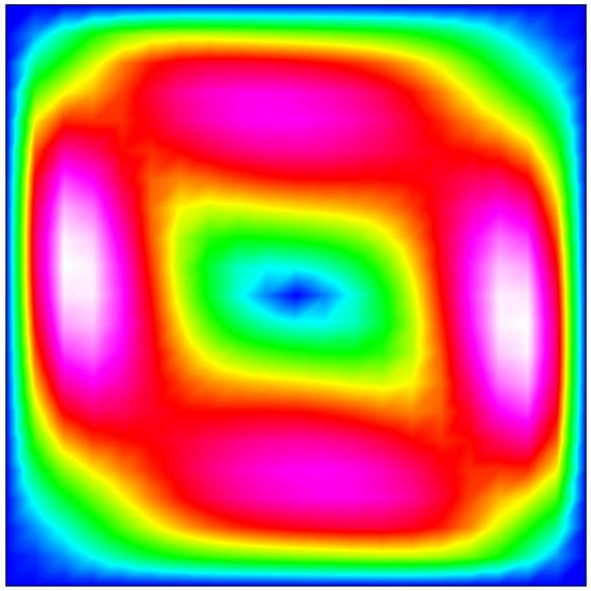




Temperature

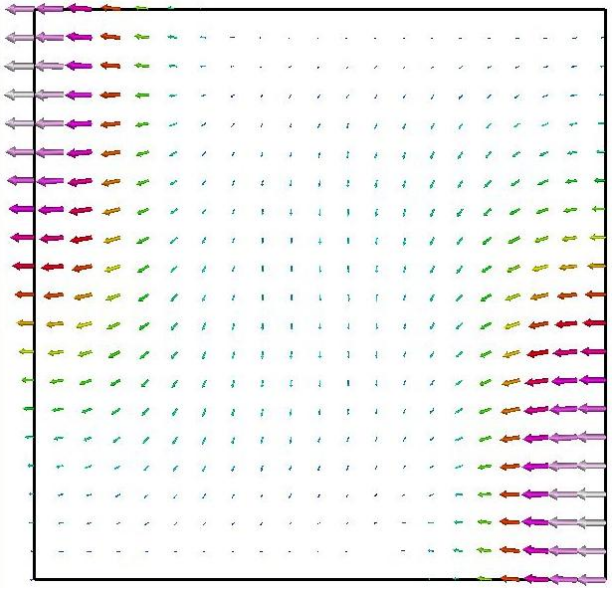


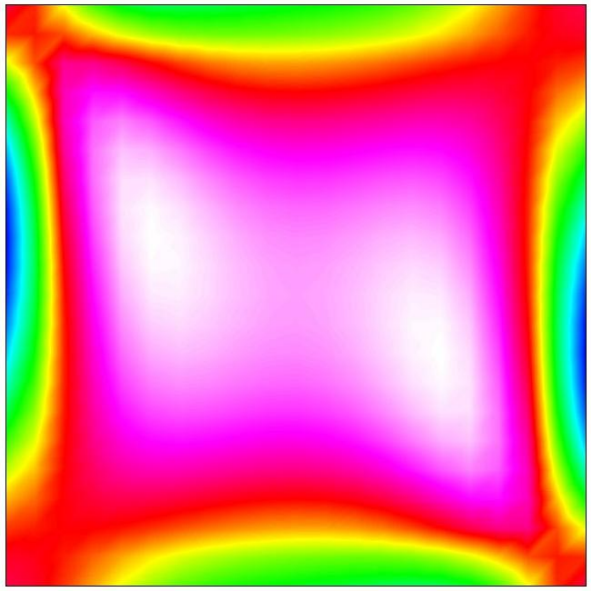
Pressure



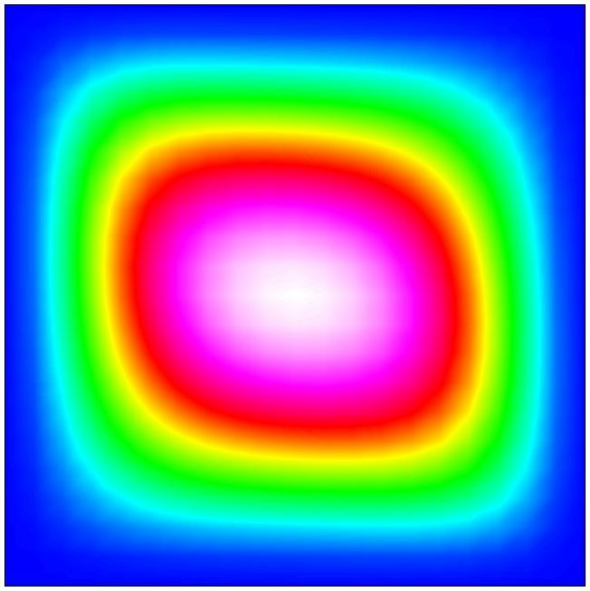
Velocity

Example: derived fields

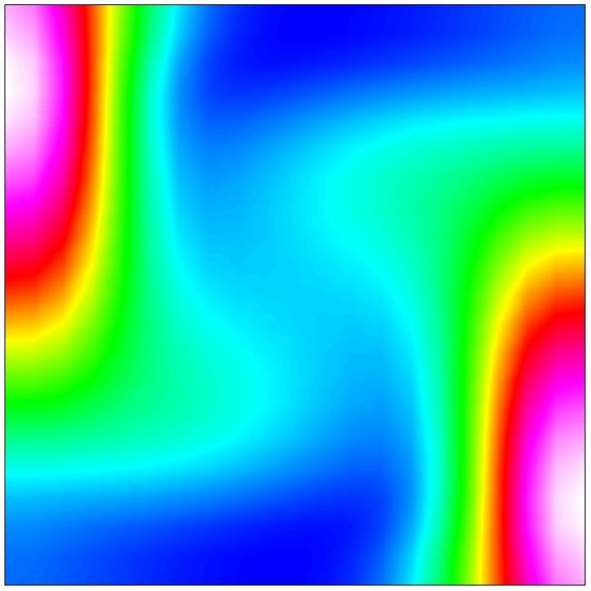




Vorticity



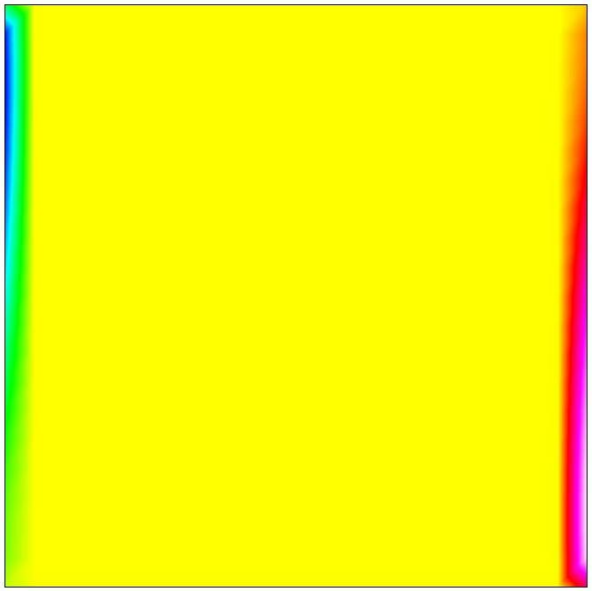
Streamlines



Diffusive flux

Example: nodal loads

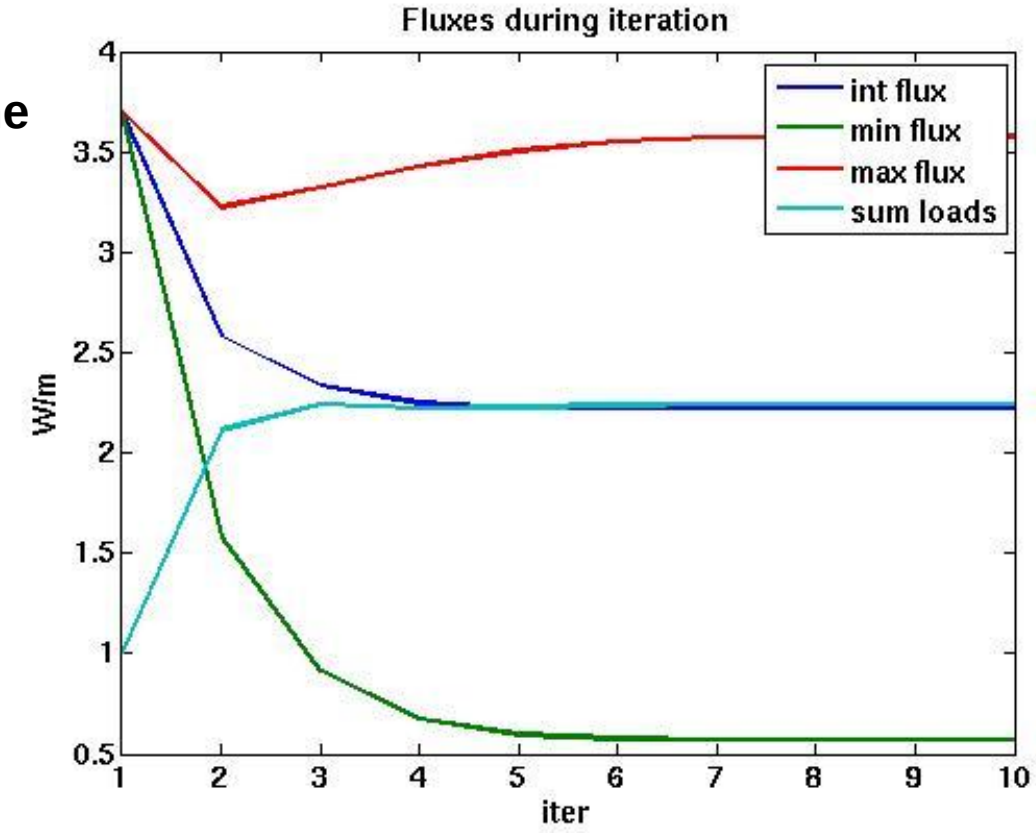
- **If equation is solved until convergence nodal loads should only occur at boundaries**
- **Element size h=1/20 ~weight for flux**



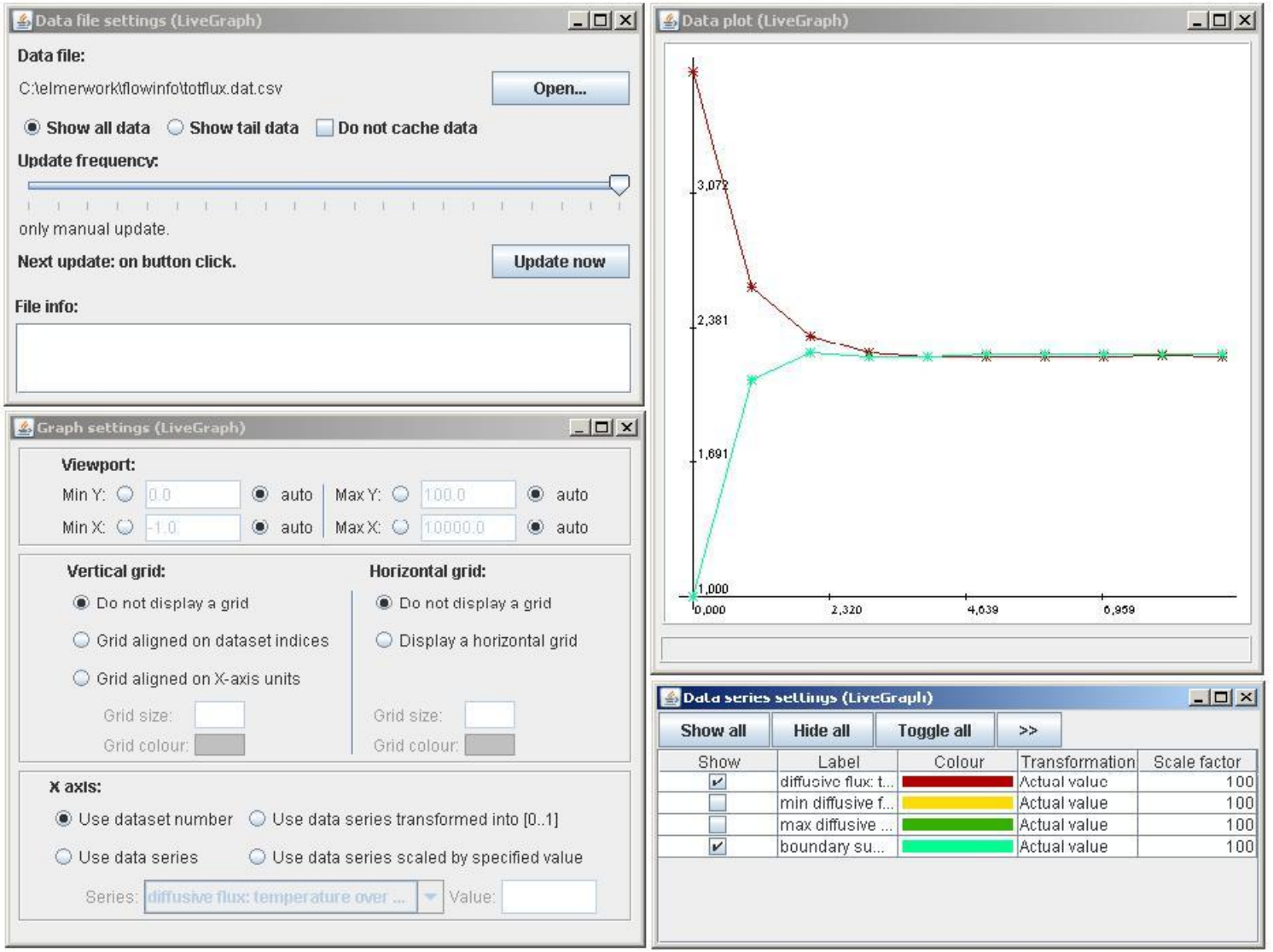
Nodal heat loads

Example: total flux

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



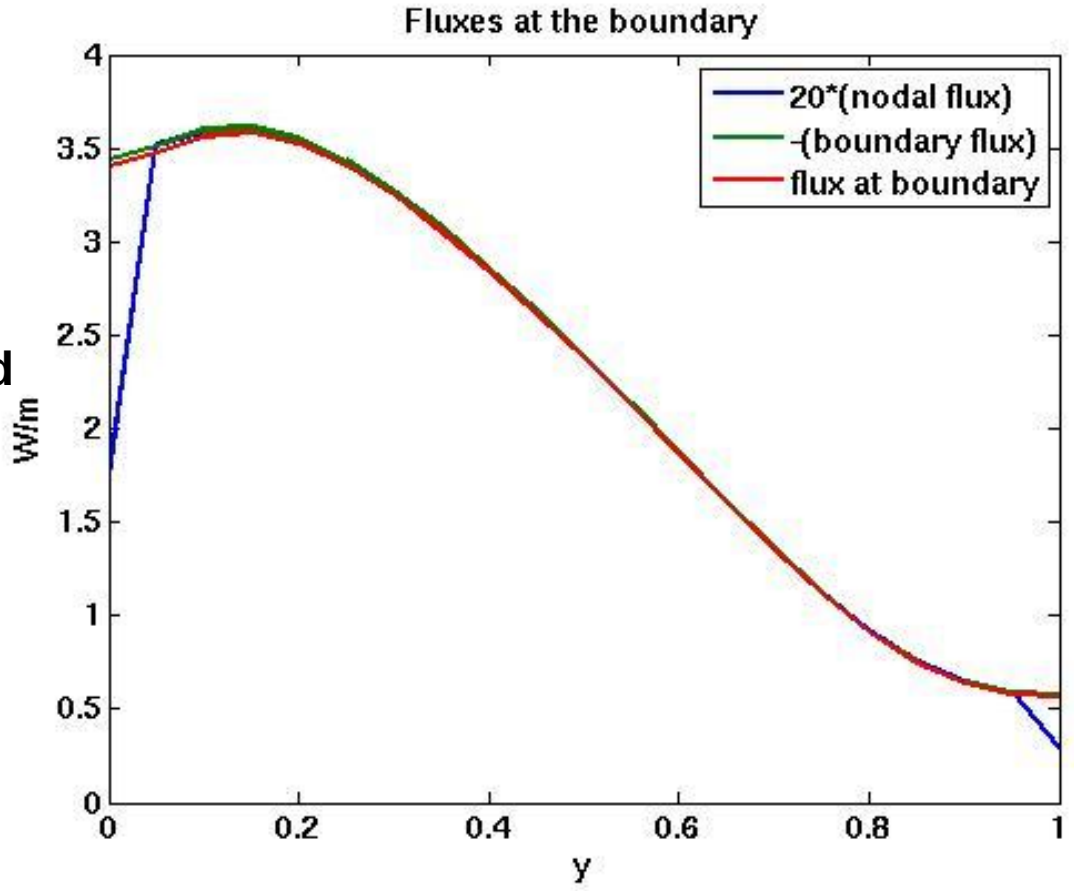
Example: view in LiveGraph (result monitoring)



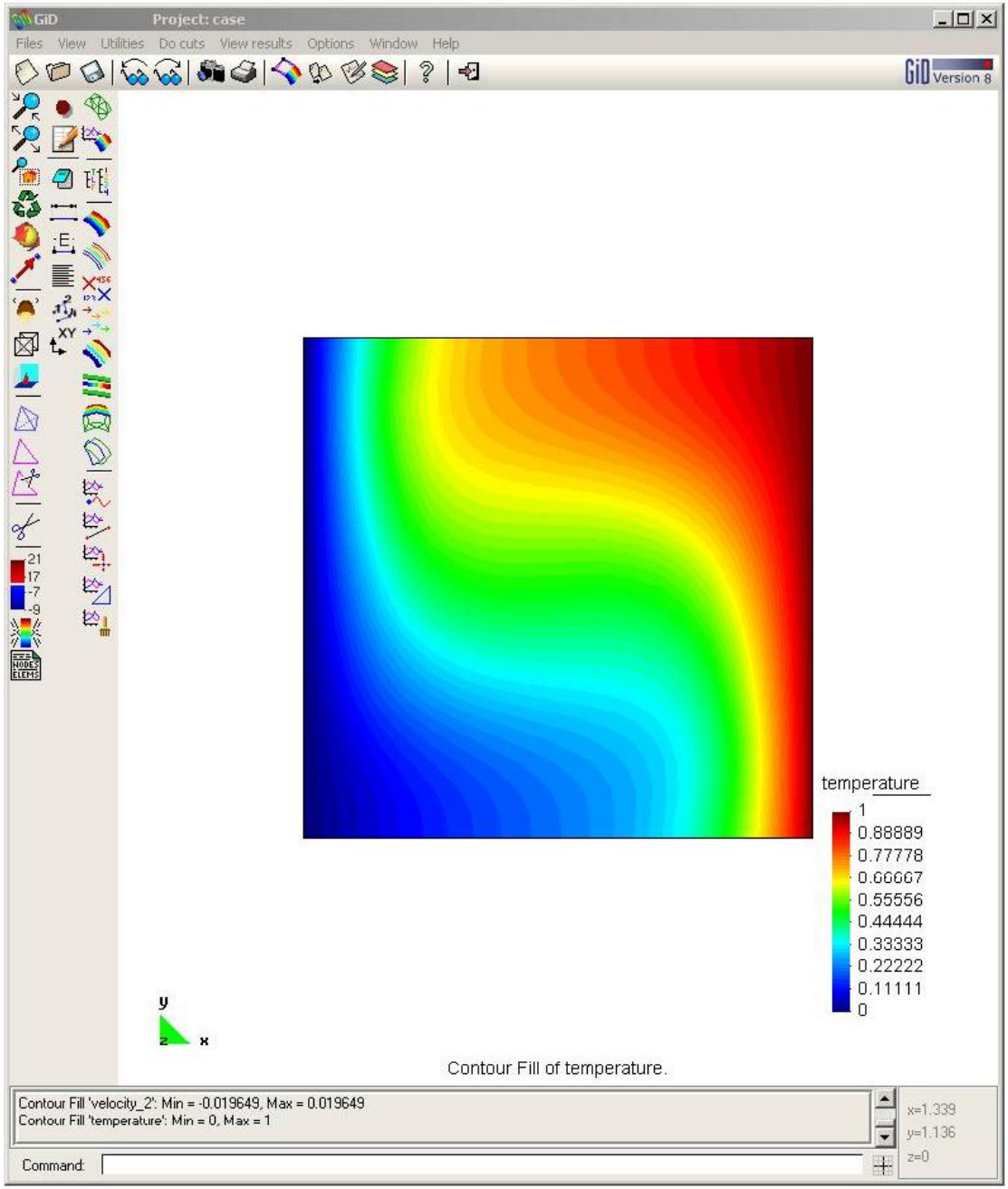
CSC

Example: boundary flux

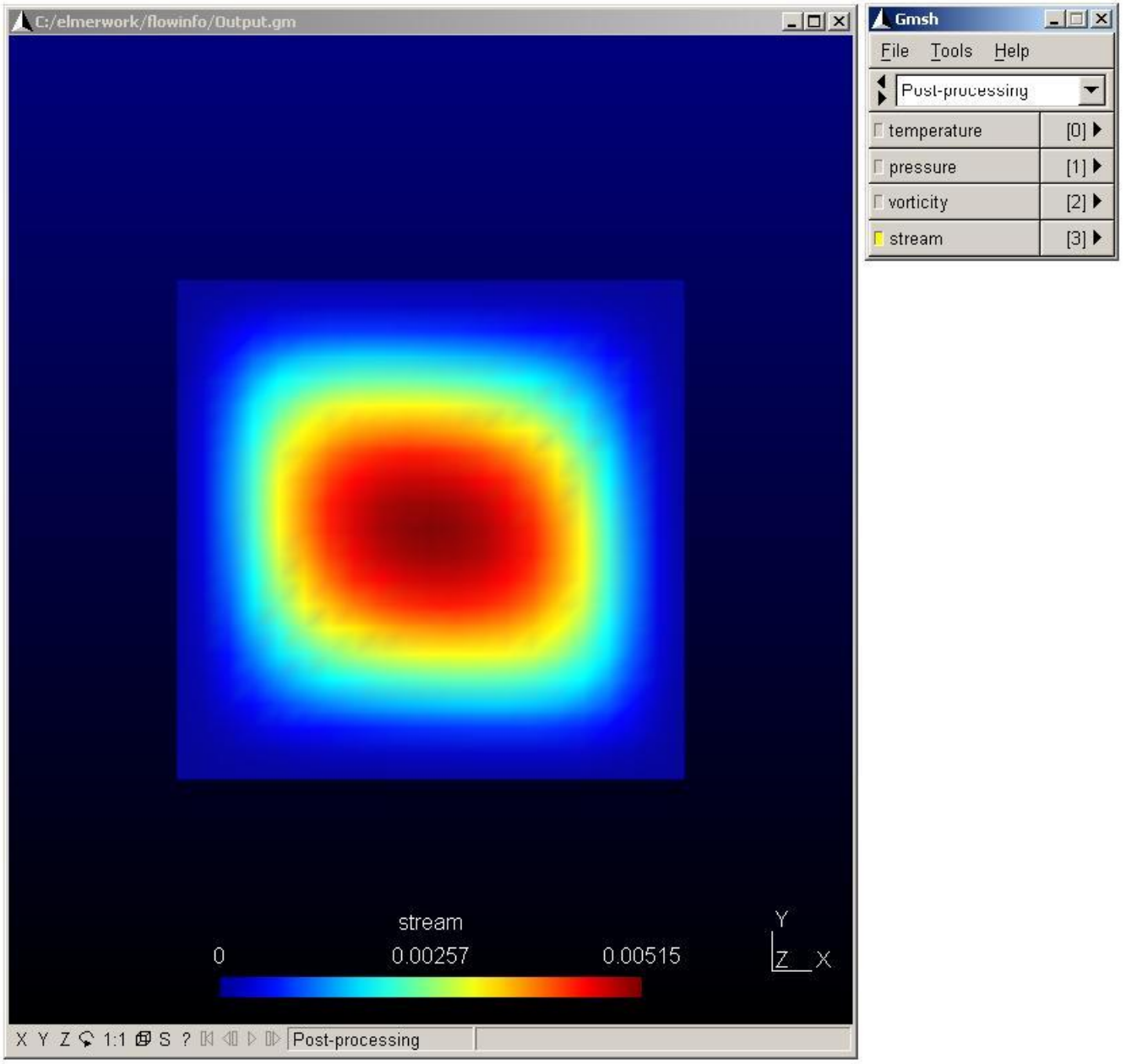
- **Saved by SaveLine**
- **Three ways of computing the boundary flux give different approximations**
- **At the corner the nodal flux should be normalized using only *h/2***



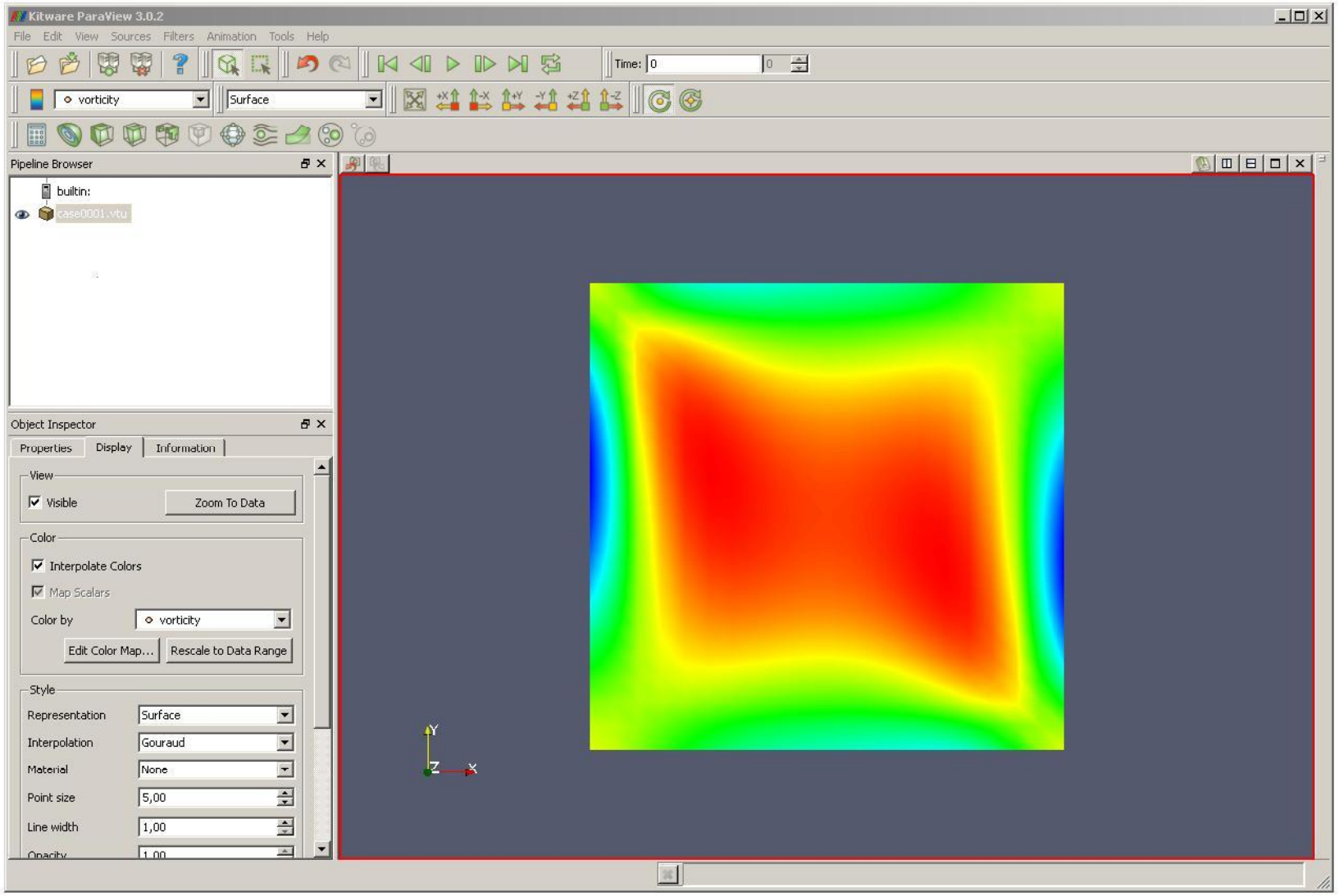
Example: view in GiD



Example: view in Gmsh



Example: view in Paraview



Alternative postprocessors for Elmer

Distributed data (all OS)

- **ElmerPost**
  - Postprocessor of Elmer suite
  - Offers all basic types of visualization
- **ParaView, Visit**
  - Good generic visualization tools
  - Offer visualization of parallel data
- **GiD, Gmsh**
  - Wise choices if these are also used as a meshes
- **OpenDX**
  - Supports only some basic elementtypes
  - Cumbersome to use

Line plotting

- **Gnuplot, R, Octave, …**
  - Open Source alternatives
- **Matlab, Excel, …**
  - Propriety software
- **LiveGraph**
  - Real time monitoring
  - Based on Java -> works on all platforms