Elmer Release 9.0

Thomas Zwinger & Peter Råback
CSC – IT Center for Science, Finland

Elmer/Ice Zoom meeting

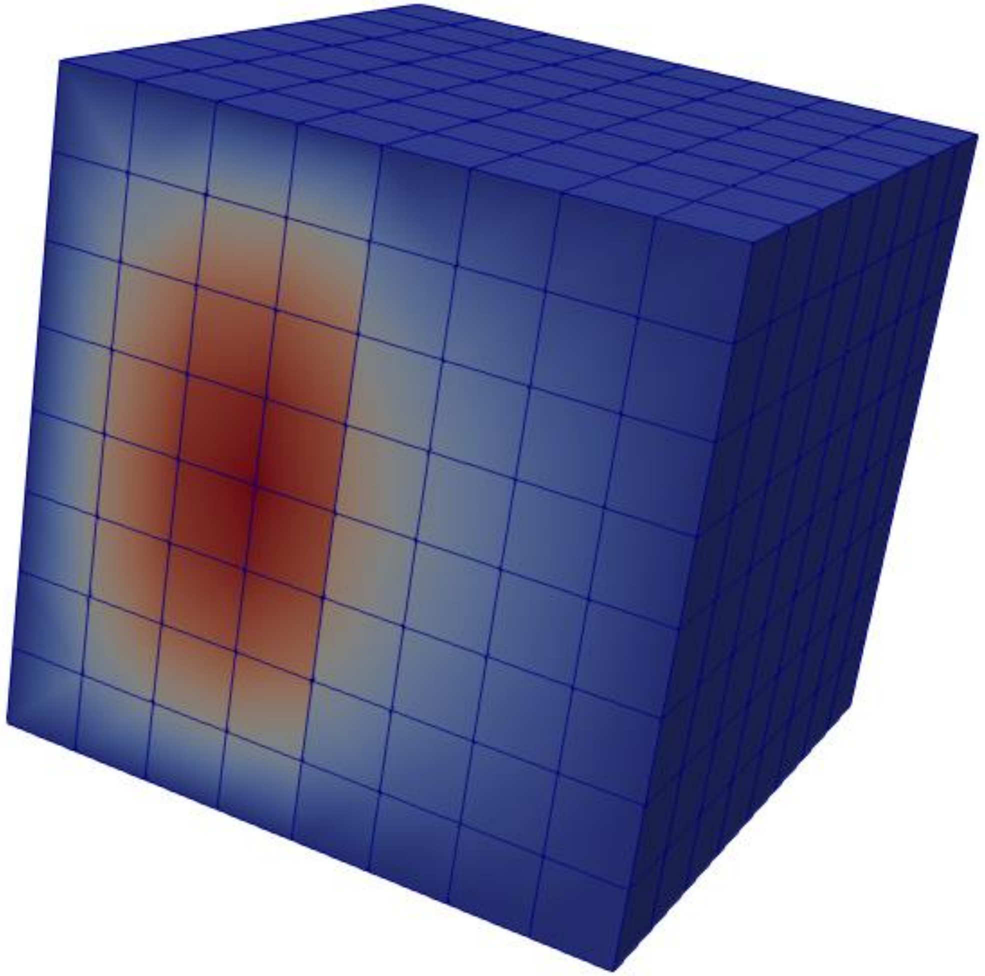
Oct 5, 2020

Release 9.0

- Why 9.0 and not 8.5?
  - There is a discontinuity in ElmerGUI
    - internal format changes from ElmerPost to VTU
    - Major faceleft thanx to Saeki!
  - And we have been several years on 8 series
- Release notes
  - ~1260 commits since last release!
  - https://github.com/ElmerCSC/elmerfem/blob/next_release/ReleaseNotes/release_9.0.md

HeatSolveVec

- Vectorized version of the old legacy modul
    - Not quite all features of the old solver available
- Some completely new functionality
    - Discontinuous Galerkin
    - Reduced Basis DG
        - Limits discontinuities between desired bodies
        - No need to manipulate the mesh
- Possible use in Elmer/Ice
    - Define jump condition between ice and bedroc



GmshReader & enhanced Gmsh output

- Restart in Elmer is cumbersome
  - separate mesh files + result files
  - Result format not used by any other software
- Sometimes we want to use save/load data only on boundaries
  - Complete restart an overkill
- Use Gmsh format
  - Same file includes mesh + results
  - Only limited functionality supported
  - Format supported by other software
  - GmshReader does interpolation on-the-fly
  - Output can be masked, e.g. some boundaries only
- Perhaps there could be use in Elmer/Ice workflows
  - Not parallel yet!

Internal partitioning

- Master partition does all the work and distributes the mesh
- Either geometric partitiong of Zoltan graph partitioning (thanx Joe!) available
- The idea is to make parallel partitioning more simple
    - Partition Mesh = Logical True + number of MPI tasks
- Also provides added flexibility
    - Various hybrid partitioning stategies possible
    - Sif may be used to determine optiomal partitioning – physics aware partitioning
- The initial partitioning is a simplification of the repartitioning needed in calving

Conforming BCs

- For confoming BCs we may identify nodes to have same (or opposite) value
- Dealt with permutation, no projection
- Reduces system size instead of increasing it
  - System size N-M
- Could be usefull for some simple benchmarks mainly
  - Faster computation

Primary solver calling other solvers

- Sometimes the nested loops of Elmer are not sufficiently flexible
- This feature enables other Solvers to be called in different stages of the primary Solver
  - Pre Solvers – before
  - Post Solvers – after
  - Nonlinear Pre Solvers – before each nonlinear iteration
  - Nonlinear Post Solvers – after each nonlinear iteration
- Together with enhanced block strategies enables solution of strongly coupled problems
  - Fluid-structure interaction, solid-shell coupling

Parametrisized runs – ”Run Control” section

- Enables outer level control of ElmerSolver run
  - Sweep over parameter space using tabulated values in external file (e.g. Dakota)
  - Internal optimization
  - And beyond
- Add’s flexibility when designing how to run large number of cases
  - Each job has small constant overhead
  - Thousands of very small jobs not always optimal
- Parameters available in sif file as
  MATC vector **rpar(0:n-1)**

Parametrisized runs – ”Run Control” section

```
Run Control
! Run predefined Dakota cases no 11-15
  Run Control Iterations = 5
  Parameter File = "LHS_distributions.out"
  Parameter Filetype = "dakota"
  Parameter Count = Integer 4
  Parameter Row Offset = 10
End

Simulation
  ! We reassign the parameters so that the sif file is nicer to read.
  $cAl=rpar(0)
  $cFe=rpar(1)

Material 3
  Name = "XAL"
  Electric Conductivity = $cAl*2.46161e+07
```

Parametrisized runs – inline parameters

- Since Fortran2008 the inline parameters are treated in a standard manner
  - Standard can be utilized since some years
- New command line argument –rpar
  - Followed by number of parameters + the parameters
  - -rpar 2 0.7 0.8
- Available in sif file as MATC vector rpar(0:n-1)
- Similarly integer arguments -ipar

Parametrisized runs – inline parameters

```
ElmerSolver one.sif –rpar 2 0.9 0.7
```

```
#!/bin/bash
for i in `seq 1 6`; do
  echo "Running Case $i"
  ElmerSolver one.sif -rpar 2 `head -$i params.dat | tail -1`
done
```

```
1.0  1.0
1.1  1.0
1.0  1.1
1.1  1.1
0.9  1.0
1.0  0.9
0.9  0.9
```

```
Simulation
! We reassign the parameters so that the sif file is nicer to read.
  $cAl=rpar(0)
  $cFe=rpar(1)

Material 3
  Name = "XAL"
  Electric Conductivity = $cAl*2.46161e+07
```

Use of environmental variables

```
#!/bin/bash
 for i in {1..5..1}
 do
    echo "Running row $i"
    export ACTIVE_ROW=$i
    ElmerSolver case.sif
 done
```

In the actual sif file we read in the row, see

```
$row=env("ACTIVE_ROW")
```

Summary of parametric operation

Built in ways to tune parameters

1. "Run Control" section + files or rules
   - Allows many cases within one simulation
   - May save time + allows internal optimization
2. Command-line arguments
   - More simple operation, external tool sets parameters
3. Environment variables
   - An alternative way to command-line arguments

$\left.\begin{array}{l}1.\ 2.\end{array}\right\}$ **MATC rpar()**

Some external parser

- Many tools automate treatmeant of input files