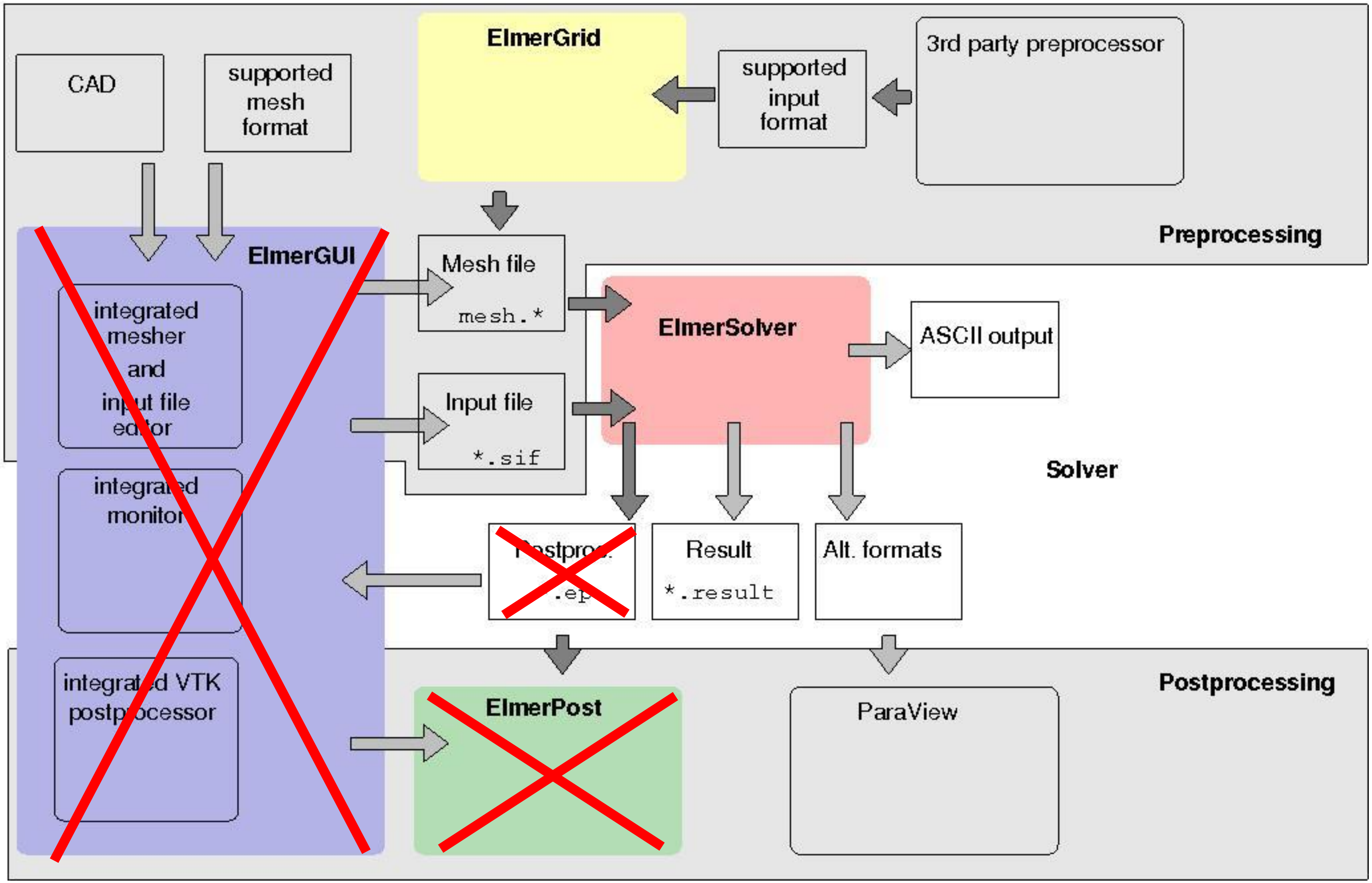
How does it work ?

Elmer structure



Sequence of a serial simulation

- build a mesh in Elmer format, i.e. a directory containing
  `mesh.header, mesh.nodes, mesh.element, mesh.boundary`

- fill in a solver input file (`mysif.sif`)

- compile object files linked with Elmer of your user functions and solvers (if needed)

- Execute :

$ **`ElmerSolver mysif.sif`**

- Should create a *.vtu files (output files in vtu format)

- Visualise :

$ **`paraview`**

We will see

- how to construct a simple mesh
- what is the content of a sif file
- how to execute
- how to visualise the results

How to get a mesh ?

Different possibilities to get a mesh

- use **ElmerGrid** alone
  - Very simple structured mesh

- use **another mesher** (gmsh, gambit) and then transform it in Elmer format - ElmerGrid can do this for many other mesh formats (just launch ElmerGrid without any argument to get list)

- Glacier particularities :
  - Small aspect ratio (horizontally elongated elements)
  - In 3D, mesh a footprint with an unstructured mesh, and then vertically extrude it (externally or internally)

will see this later during the course…

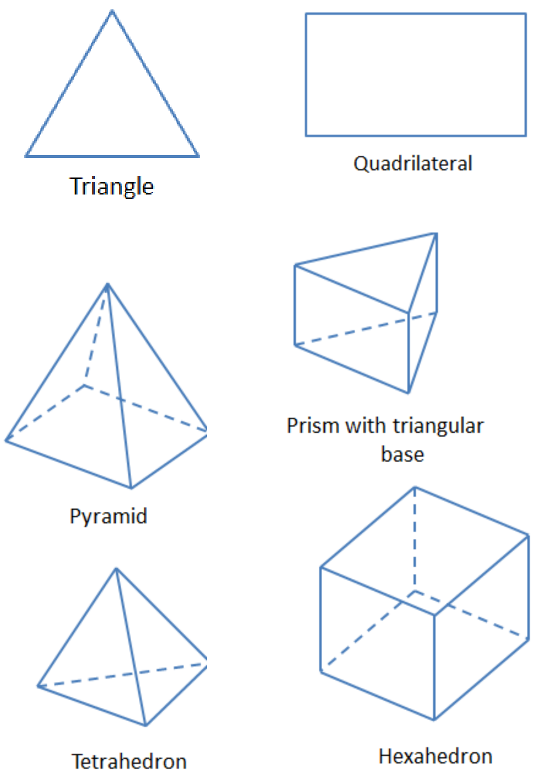
ElmerGrid

- command line tool for mesh generation
- native mesh format: `.grd`
- help : just execute : `$ ElmerGrid`
- possible to import meshes produced by other free or commercial mesh generators (UNV, Comsol, **gmsh**, …)
- Examples :

```
$ ElmerGrid  1 2 my_mesh.grd
$ ElmerGrid 14 2 my_gmsh_mesh.msh -autoclean
$ ElmerGrid 14 5 my_gmsh_mesh.msh -autoclean
```

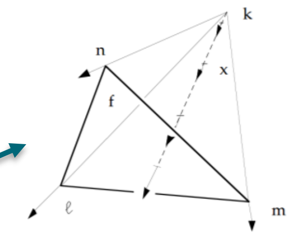
Elmer mesh – Finite element shapes

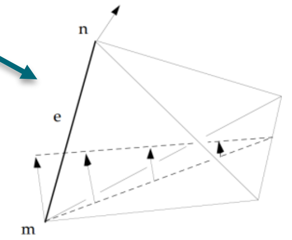
- All standard shaper of Finite Elements are supported
  - 0D: point
  - 1D: segment
  - 2D: triangles, quadrilaterals
  - 3D: tetraherdons, wedges, pyramids, hexahedrons
- Meshes may have mixed element types
- There may be also several meshes in same simulation



Elmer mesh – basis functions

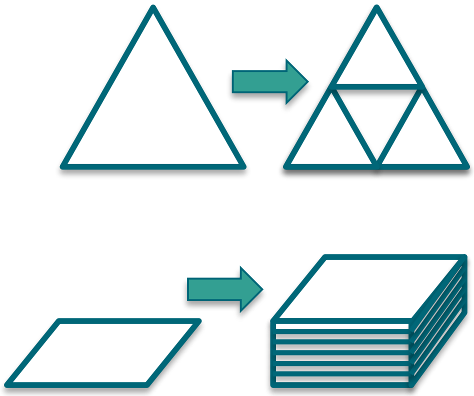
- Element families
  - Nodal (up to 2-4th degree)
  - p-elements (up to 10th degree)
  - Edge & face –elements
    - H(div) - often associated with"face" elements)
    - H(curl) - often associated with "edge" elements)
- Formulations
  - Galerkin, Discontinuous Galerkin
  - Stabilization
  - Residual free bubbles





Elmer mesh – internal mesh generation

- Internal mesh division
  - 2^*DIM*^*n* -fold problem-size
  - Known as "**Mesh Multiplication**"
  - Simple inheritance of mesh grading
- Internal mesh extrusion
  - Extruded given number of layers
- Idea is to remove bottle-necks from mesh generation
  - These can also be performed on a parallel level
- Limited by generality since the internal meshing features cannot increase the geometry description



Solver Input File (sif)

Example of sif file

- Comments start with !
- Not case sensitive
- Avoid non-printable characters (e.g., tabulators for indents)

- A section always ends with the keyword `End` or use `::`

- Parameters not in the Keyword DB need to be casted by types:
  `Integer`, `Real`, `Logical`, `String` and `File`

- `Paremetername(n,m)` indicates a n × m array

- Sections are

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

```
Body Force 1
Heat Source = 1.0
End
```

OR

```
Body Force 1 :: Heat Source = 1.0
```

Example of sif file

```
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!                                       !!
!! Elmer/Ice Course - Application Step0  !!
!!                                       !!
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
! Updated May 2011

check keywords warn
echo on

Header
  Mesh DB "." "square"
End

Constants
! No constant needed
End

!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
Simulation
  Coordinate System  = Cartesian 2D
  Simulation Type = Steady State

  Steady State Min Iterations = 1
  Steady State Max Iterations = 1

  Output File = "ismip_step0.result"
  Post File = "ismip_step0.vtu"
  max output level = 100
End

!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
Body 1
  Equation = 1
  Body Force = 1
  Material = 1
  Initial Condition = 1
End

!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
Initial Condition 1
  Pressure = Real 0.0
  Velocity 1 = Real 0.0
  Velocity 2 = Real 0.0
End

!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
Body Force 1
  Flow BodyForce 1 = Real 0.0
  Flow BodyForce 2 = Real -1.0
End
```

- **Header** declares where to search for the mesh
- If any **constants** needed (i.e. Gas constant)
- **Simulation**
  - Type of coordinate system
  - Steady or Transient
    - If transient: time stepping parameters
  - Output files (to restart a run) and VTU file
  - Output level : how verbose is the code?
  - Restart information (optional)
- In **Body** are assigned the Equation, Body Force, Material and Initial Condition
- In **Initial Condition** sets initial variable values
- In **Body Force** specify the body force entering the right side of the solved equations

Example of sif file

```
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
Material 1
  Density = Real 1.0

  Viscosity Model = String "power law"
  Viscosity = Real 1.0
  Viscosity Exponent = Real 0.3333333333333333
  Critical Shear Rate = Real 1.0e-10
End

!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
Solver 1
  Equation = "Navier-Stokes"

  Stabilization Method = String Bubbles
  Flow Model = String Stokes

  Linear System Solver = Direct
  Linear System Direct Method = umfpack

  Nonlinear System Max Iterations = 100
  Nonlinear System Convergence Tolerance  = 1.0e-5
  Nonlinear System Newton After Iterations = 5
  Nonlinear System Newton After Tolerance = 1.0e-02
  Nonlinear System Relaxation Factor = 1.00

  Steady State Convergence Tolerance = Real 1.0e-3
End

!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!

Equation 1
  Active Solvers(1)= 1
End

!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
Boundary Condition 1
  Target Boundaries = 1
  Velocity 2 = Real 0.0e0
End

Boundary Condition 2
  Target Boundaries = 4
  Velocity 1 = Real 0.0e0
End

Boundary Condition 3
  Target Coordinates(1,2) = Real 0.0 1.0
  Target Coordinates Eps  = Real 1.0e-3
  Pressure = Real 0.0e0
End
```

- In **Material** sets material properties for the body (can be scalars or tensors, and can be given as dependent functions)
- In **Solver** specifies the numerical treatment for these equations (methods, criteria of convergence,…)
- In **Equation** sets the active solvers
- **Boundary Condition**
  - Dirichlet: `Variablename = Value`
  - Neumann: special keyword depending on the solver
  - Values can be given as function

Variable defined as a function

1) Tables can be use to define a piecewise linear (cubic) dependency of a variable

```
Density = Variable Temperature
Real cubic
   0  900
 273 1000
 300 1020
 400 1000
End
```

Outside range: Extrapolation!

2) MATC: a library for online (in SIF file) numerical evaluation of mathematical functions

```
Density = Variable Temperature
MATC "1000*(1 - 1.0e-4*(tx-273.0))"
```

Evaluated every time

or as constant expressions

```
Viscosity Exponent = Real $1.0/3.0
```

Evaluated once

3) Build your own user function

```
Density = Variable Temperature
  Procedure "filename" "proc"
```

filename should contain a shareable (.so on Unix) code for the user function whose name is `proc`

Example of User Function

in the filename.F90 file :

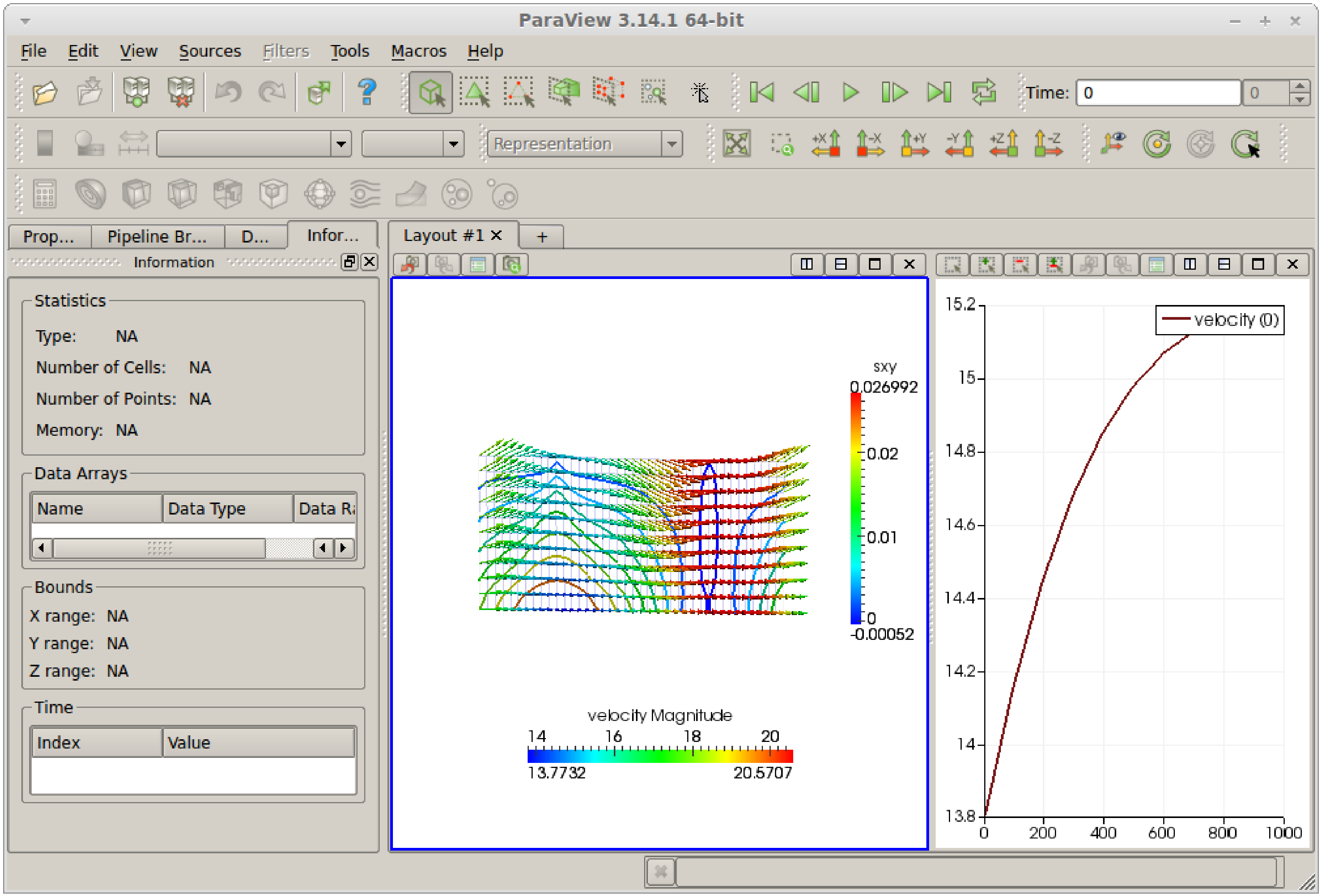
```
FUNCTION proc( Model, n, T ) RESULT(dens)
USE DefUtils
IMPLICIT None
TYPE(Model_t) :: Model
INTEGER :: n
REAL(KIND=dp) :: T, dens

    dens = 1000*(1-1.0d-4 *(T-273.0_dp))

END FUNCTION proc
```

Compilation tools: `elmerf90`

`$` **`elmerf90`** ***`filename`*** **`.F90 -o`** ***`filename`*** **`.so`**

How to visualise results

Paraview


ParaView 3.14.1 64-bit
File Edit View Sources Filters Tools Macros Help
Time: 0
Representation
Prop... Pipeline Br... D... Infor...
Layout #1
Information
Statistics
Type: NA
Number of Cells: NA
Number of Points: NA
Memory: NA
Data Arrays
Name Data Type Data Ra
Bounds
X range: NA
Y range: NA
Z range: NA
Time
Index Value
sxy
0.026992
0.02
0.01
0
-0.00052
velocity Magnitude
14 16 18 20
13.7732
20.5707
velocity (0)
15.2
15
14.8
14.6
14.4
14.2
14
13.8
0 200 400 600 800 1000

ASCII Based Output

`SaveScalars`	e.g. CPU time, mean, max, min of a variable, Flux
`SaveLine`	save a variable along a line (boundary or a given line)
`SaveMaterials`	save a material parameter like a variable

Example:

```
Solver 3
  Exec Solver =  After All
  Procedure = File "SaveData" "SaveLine"
  Filename =  "ismip_surface.dat"
  File Append = Logical False
End


Solver 4
  Exec Solver =  After TimeStep  ! For transient simualtion
  Procedure = File "./MySaveData" "SaveScalars"
  Filename =  "ismip_scalars.dat"
  File Append = Logical True     ! For transient simualtion

  Variable 1 = String "Flow Solution"
  Operator 1 = String "Volume"

  Variable 2 = String "Velocity 1"
  Operator 2 = String "Max Abs"

  Variable 3 = String "Flow Solution"
  Operator 3 = String "Convective flux"

  Variable 4 = String "cpu time"

  Variable 5 = String "cpu memory"
End
```

```
! Upper Surface
Boundary Condition 3
  Target Boundaries = 3
  Save Line = Logical True
  Flux integrate = Logical True
End
```

Good to know

- The structure of sif file has almost one-to-one mapping with Model type and its lists
  - Each keyword is an entry in list structure
- For many tasks there exists a separate solver a.k.a. module
  - Don't be afraid to add new addition solvers
  - Elmer modules + Elmer/Ice solvers
- Copy-paste works is often a good way to start
  - Hundreds of consistency tests under elmerfem/fem/test and elmerice/Tests
- Documentation is never complete – ask!