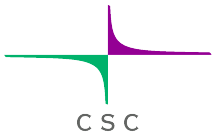


Elmer/Ice course

31st Oct, 1st & 2nd Nov 2016, Oslo

Introduction

Thomas ZWINGER (1)

Fabien GILLET-CHAULET(2)

Adrien GILBERT(3)

Olivier GAGLIARDINI (2)

(1) CSC - IT Center for Science Ltd. – Espoo - Finland
(2) LGGE - Grenoble - France
(3) University of Oslo - Oslo - Norway

Program

Monday 31st October

9:00-9:30 Arrival of the participants at University of Oslo

9:30-9:45 Welcome words by Thomas Schuler, general announcements

9:45-10:30 Introduction Elmer/ice (OG)

10:30-11:00 Coffee break

11:00-12:00 Toy flow-line model: basic diagnostic (TZ)

12:00 Lunch (sandwiches)

13:00-15:30 Toy flow-line model: thermo-mechanical coupling, sliding, prognostic runs (TZ)

15:30-16:00 Coffee break

16h00-17h30 Tête Rousse context and diagnostic (OG)

Program

Tuesday 1st November

9:00-10:30 Tête Rousse setup and diagnostic (OG)

10:30-11:00 Coffee break

11:00-12:00 Tête Rousse prognostic (OG)

12:00 Lunch (sandwiches)

13:00-14:30 Coupling ice flow and temperature (AG)

14:30-15:00 Coffee break

15:00-17:00 Coupling ice flow and temperature (AG)

19:00 - Course dinner (place to be specified)

Program

Monday 2nd November

9:00-10:30 Inverse modelling and SSA (FG)

10:30-11:00 Coffee break

11:00-12:00 Inverse modelling and SSA (FG)

12:00 Lunch (sandwiches)

13:00-14:30 Inverse modelling and SSA (FG)

14:30-16:00 Questions on your own modelling

16h00 End of the course

Short history of Elmer/Ice (not anymore so short...) 1/2

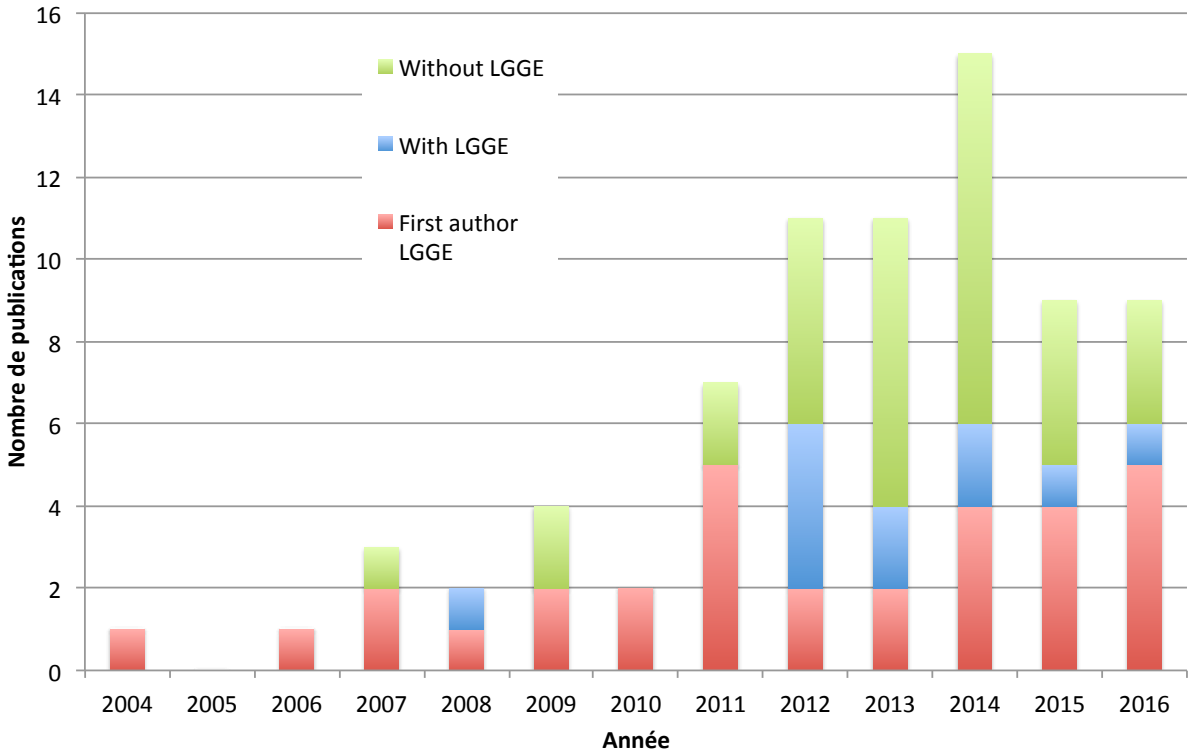
- ✓ EGU2002: OG was looking for a 3D FE code to model the flow of strain-induced anisotropic polar ice – meet TZ
- ✓ March 2003: OG visited CSC for few days: AIFlowSolver and FabricSolver partly implemented
- ✓ August 2005 – One year visit of OG at CSC (Anisotropy, cavity, glaciers, ISMIP tests, ...)
- ✓ February 2008 – First Elmer/Ice Course - Grenoble
- ✓ June 2011 – SVALI summer school – Finland
- ✓ 2012 – Elmer/Ice has now a website, a logo and a mailing list
- ✓ 2012 – Elmer/Ice comes as a Elmer Package – New wiki
- ✓ 2012 – Elmer/Ice course at UBC/SFU
- ✓ 2013 – Elmer/Ice courses at Univ. Washington and Univ. Alberta
- ✓ 9 April 2013 – First Elmer/Ice users meeting – EGU 2013

Short history of Elmer/Ice (not anymore so short...) 2/2

- ✓ May 2013 – Second SVALI summer school – Finland
- ✓ 2-day beginner Elmer/Ice course, 3-4 Oct. 2013, LGGE, Grenoble, France
- ✓ 3-day Elmer/Ice advanced workshop, 4-6 Nov. 2013, CSC, Espoo, Finland
- ✓ April 2014 – Second Elmer/Ice users meeting – EGU 2014
- ✓ 3-day beginner Elmer/Ice course, 27-29 Oct. 2014, IMO, Reykjavík, Iceland
- ✓ April 2015 – Third Elmer/Ice users meeting – EGU 2015
- ✓ 2-day beginner course, 1&2 Nov, CIC, Copenhagen, Denmark
- ✓ 3-day Elmer/Ice advanced workshop, 30 Nov, 1&2 Dec 2015, LGGE, Grenoble, France

and now Oslo...

A growing community



12 Courses ~ 220 participants

3 Elmer/Ice users meetings

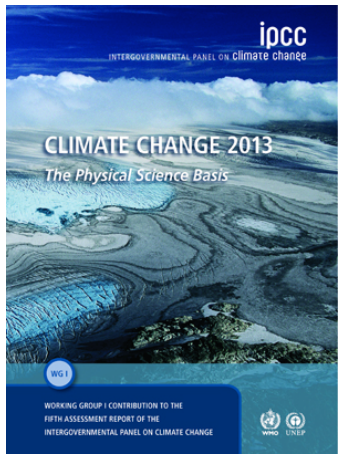
75 publications

Since few years, first authors are not only anymore only from CSC or LGGE...

Elmer/Ice applications

75 publications using Elmer/Ice since 2004

- ISMIP, MISMIP, MISMIP-3d
- 2D and 3D Grounding line dynamics
- Ice2sea and SeaRISE contributions (Greenland)
- Inverse methods (Variegated, Vestfonna ice-cap, GIS)
- Flow of anisotropic ice
- Glaciers
- 9 cited references including results from Elmer/Ice in the 5th IPCC report

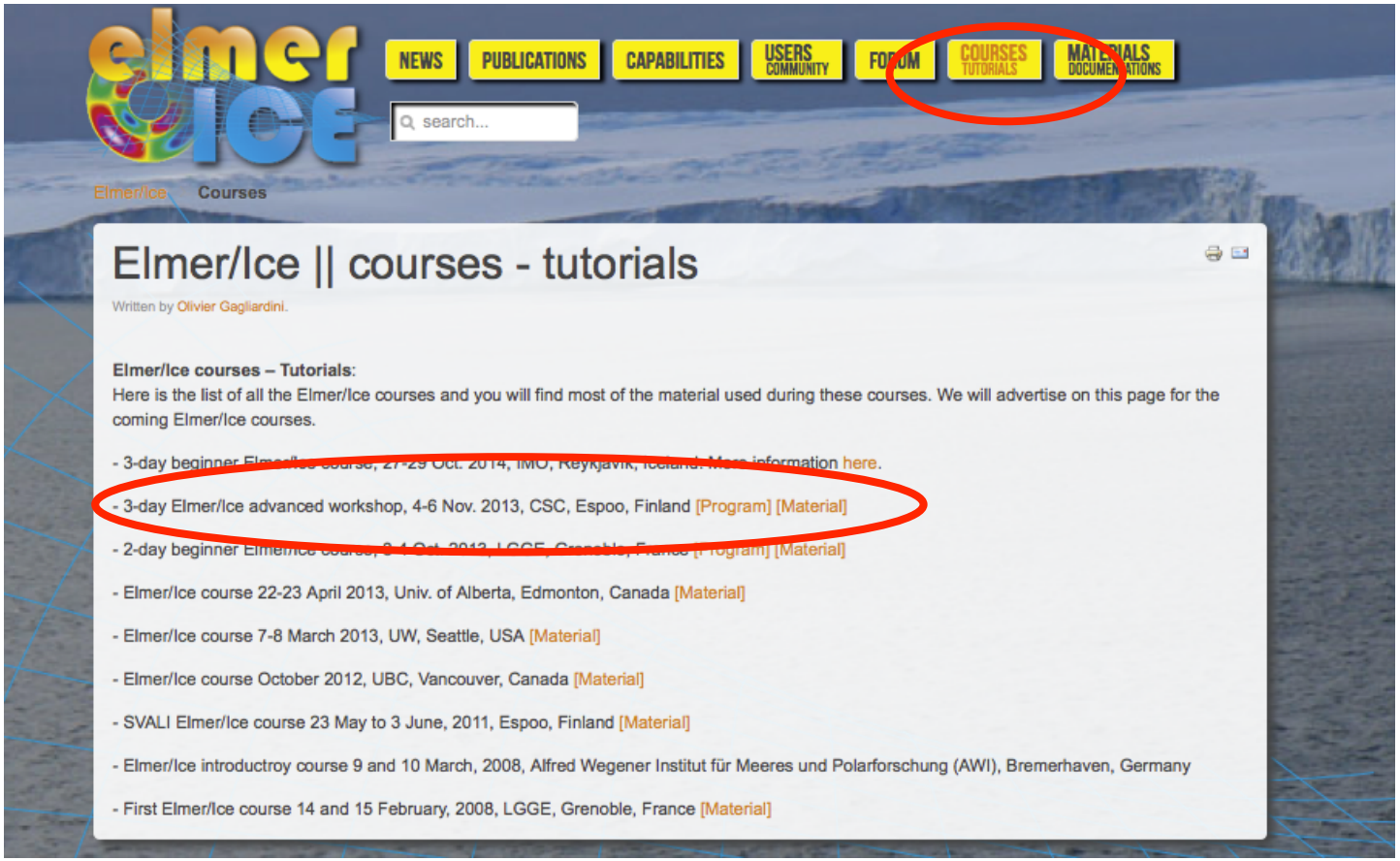


see http://elmerice.elmerfem.org/publications

Elmer/Ice website

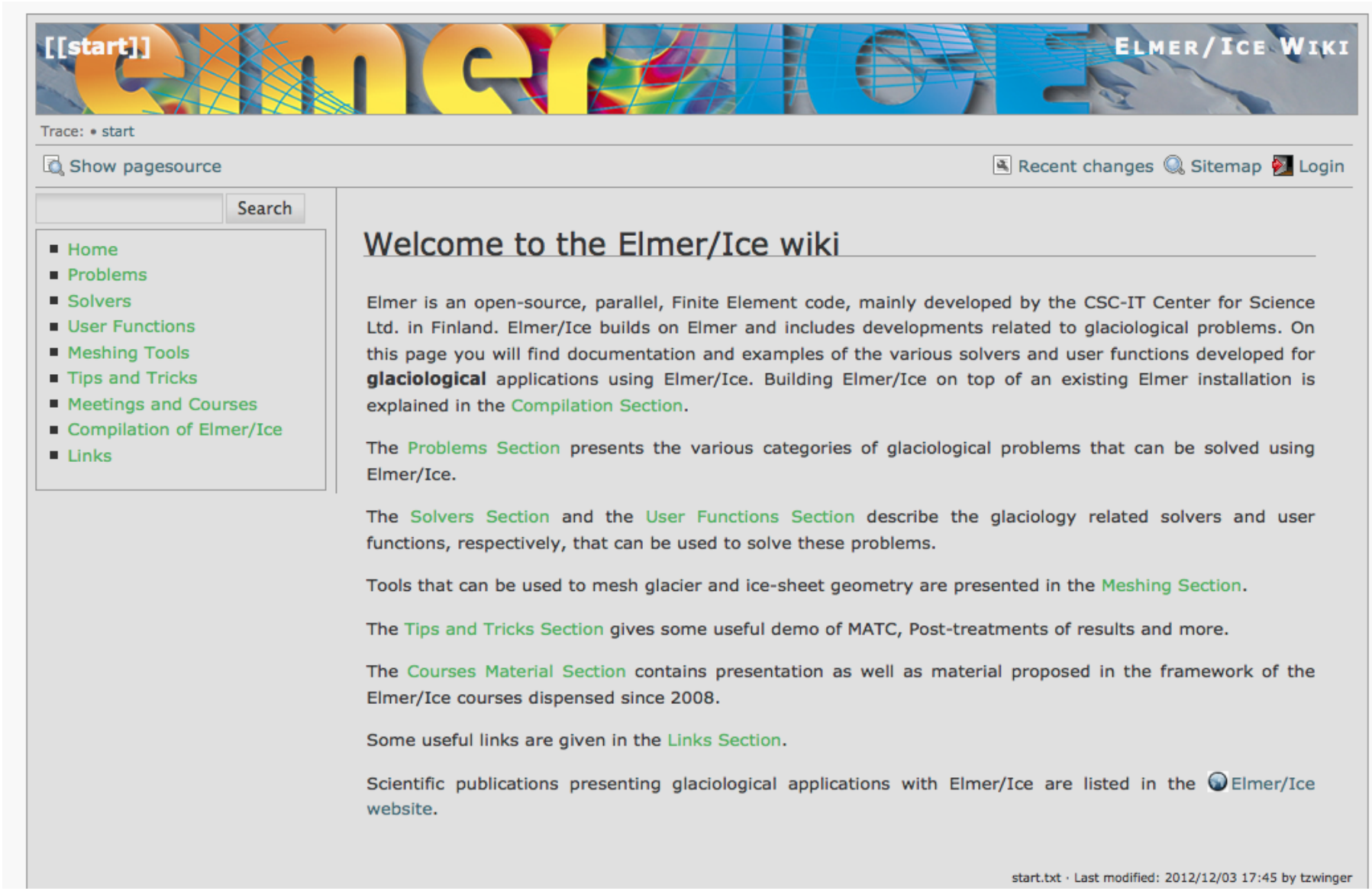
http://elmerice.elmerfem.org/

Elmer/Ice website http://elmerice.elmerfem.org/



Much more material available than what we will present today

Elmer/Ice wiki http://elmerice.elmerfem.org/wiki/doku.php



Elmer/Ice mailing list

To subscribe to the Elmer/Ice list *elmerice@elmerfem.org*, just sent an email to *majordomo@elmerfem.org*, with in the body the text:

subscribe elmerice

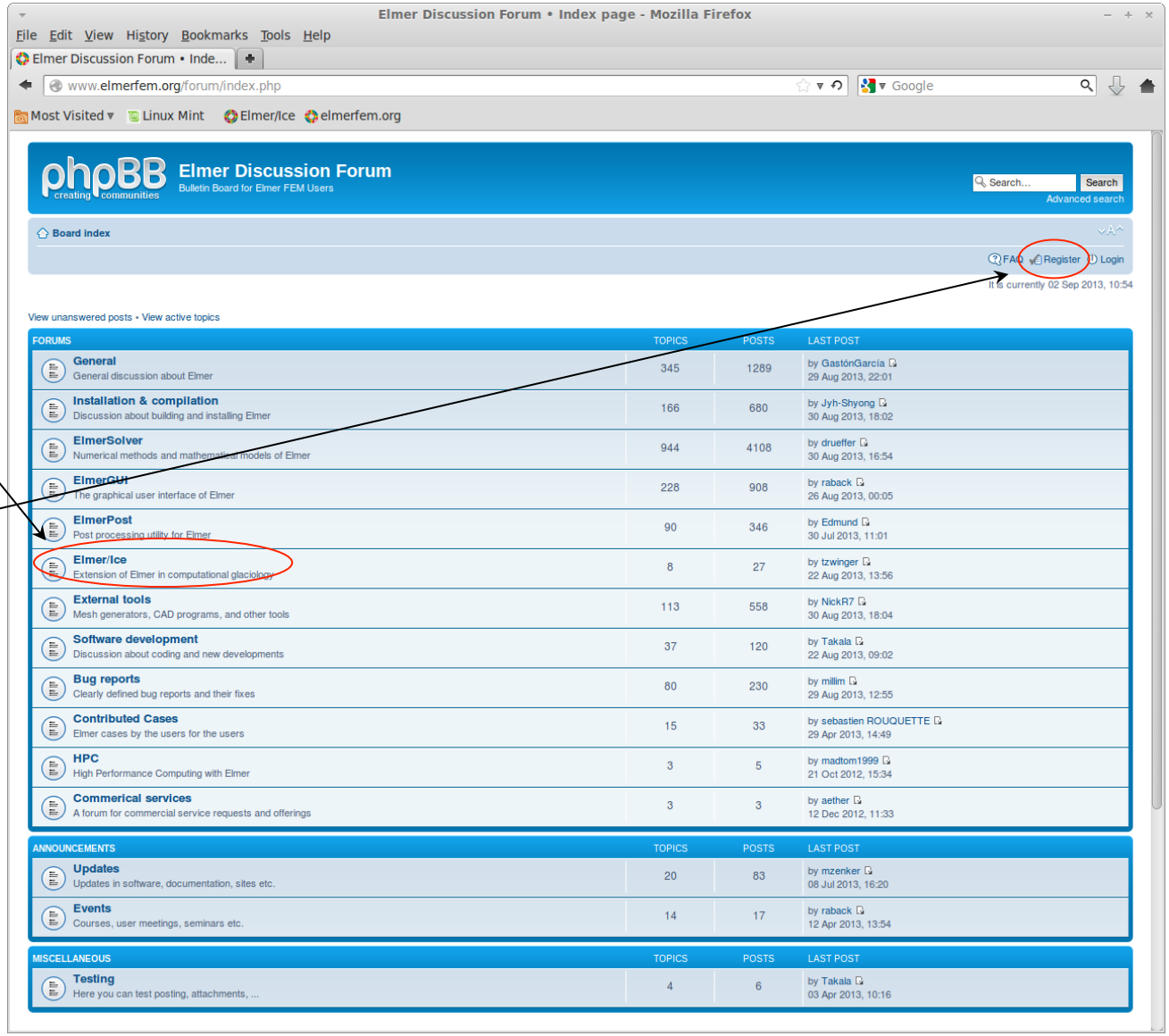
If you do not know how to use mailing lists run by majordomo you may sent a mail with "help" in the message body.

Elmer/Ice Forum

Under

http://www.elmerfem.org :

- Go to **Elmer Forum:** find answers on all aspects of Elmer
- Click on **Elmer/Ice** link: find answers specific to Elmer/Ice
- To get access: **Register** in upper right corner



Elmer/Ice on Twitter @ElmerIce1

Important links (summary)

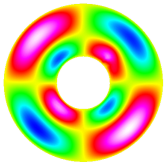
Elmer at CSC (documentation, how to install, …)
http://www.elmerfem.org/
http://www.csc.fi/english/pages/elmer

Elmer Forum
http://elmerfem.org/forum/

Elmer/Ice webpage
http://elmerice.elmerfem.org/

Elmer/Ice wiki
http://elmerice.elmerfem.org/wiki/doku.php?id=start

Elmer/Ice in relation to Elmer



Elmer is an open-source, parallel, Finite Element code, mainly developed by the CSC-IT Center for Science Ltd. in Finland.

Elmer is constantly developed towards improved performance, utilizing international projects such as FP7 PRACE and HPC Europa2.

Elmer/Ice builds on Elmer and includes developments related to glaciological problems.

Elmer/Ice includes a variety of dedicated solvers and user functions for glaciological applications and its development is supported by various groups and funding...

Elmer/Ice Package

All the Solvers, User Functions and Meshers presented on the Elmer/Ice wiki comes as an Elmer/Ice package on the Elmer distribution (in `elmerice/`)

To get Elmer/Ice installed, add the following option to the cmake build command:

`-DWITH_ElmerIce:BOOL=TRUE`

To use it (in the SIF file):

`Procedure = File "ElmerIceSolvers" "`*`NameSolver`*`"`

or

`Procedure = File "ElmerIceUSF" "`*`NameUSF`*`"`

Important notices

In this course

- We will not teach finite element method (can give references)

- We will focus on some technical aspects of using Elmer for glaciological applications

What we expect from this course ?

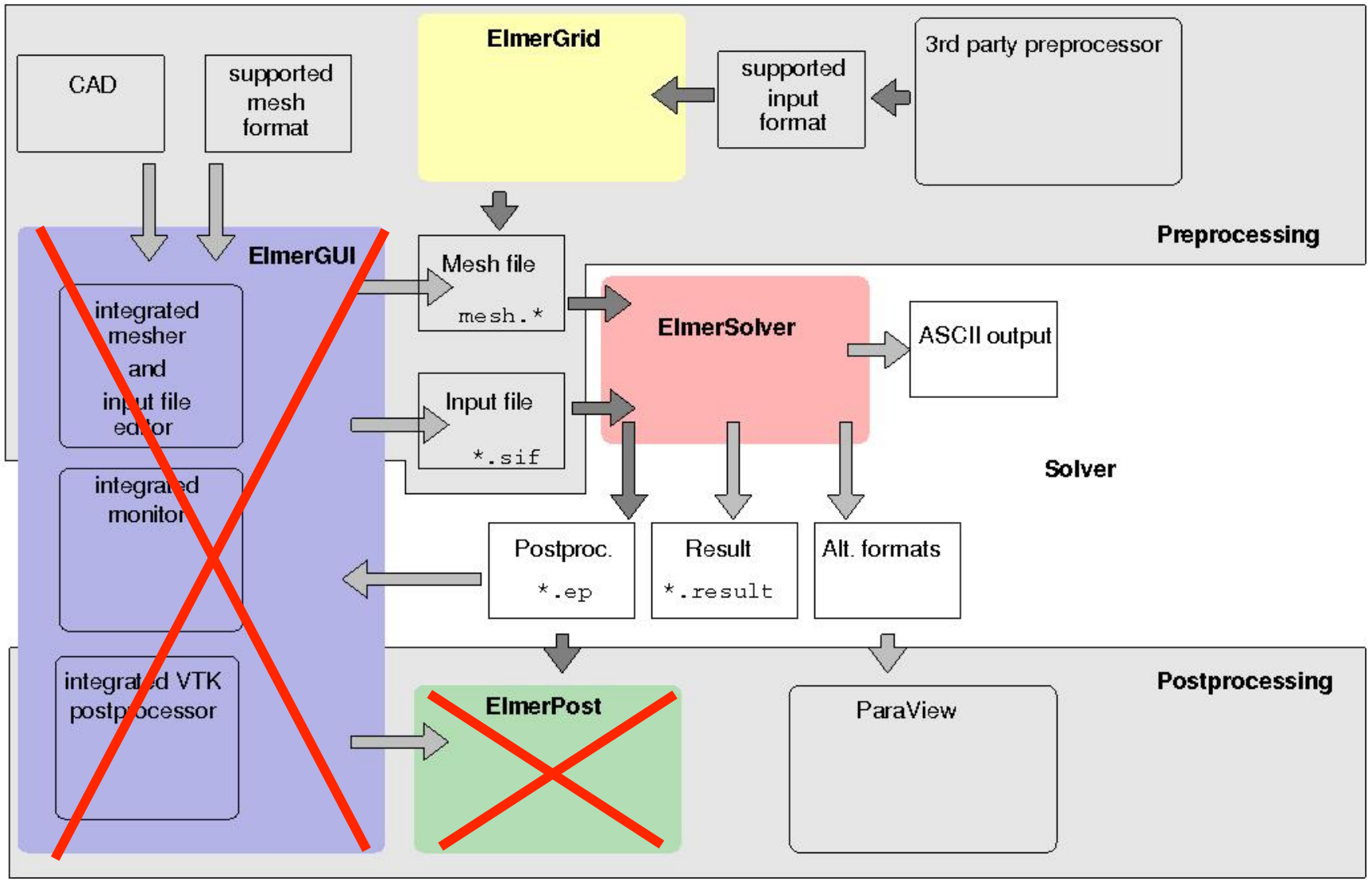
- giving you a kick-start in Elmer/Ice
- some fruitful collaborations to begin

Elmer/Ice capabilities

- **Full-Stokes** equations but also SIA, SSA, diagnostic or transient
- Various **rheologies** (Glen's law, firn/snow and anisotropic flow laws)
- **Temperature** solver accounting for the upper limit at melting point
- **Transport equations** for density, fabric, age …
- **Post-processing solver** for strain-rate and stress fields
- Various **friction laws** (Weertman, effective-pressure dependent friction law)
- **Free surface evolution** as a contact problem (Grounding line dynamics)
- **Inverse methods** (linear adjoint and Arthern and Gudmundsson 2010 methods)
- Tools or plug-ins for **meshing** (YAMS, external and internal extrusion of footprint)
- **Highly parallel** Stokes solver
- **Basal hydrology** (2 approaches, one in the distribution)
- **Calving** (3 approaches, not yet in the distribution)
- **Damage mechanics**

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

- 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 (Ansys, Abaqus, Fluent/Neutral, 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
```

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 equation

Example of sif file

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

  Viscosity Model = String "power law"
  Viscosity = Real 1.0
  Viscosity Exponent = Real 0.333333333333333333
  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) 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

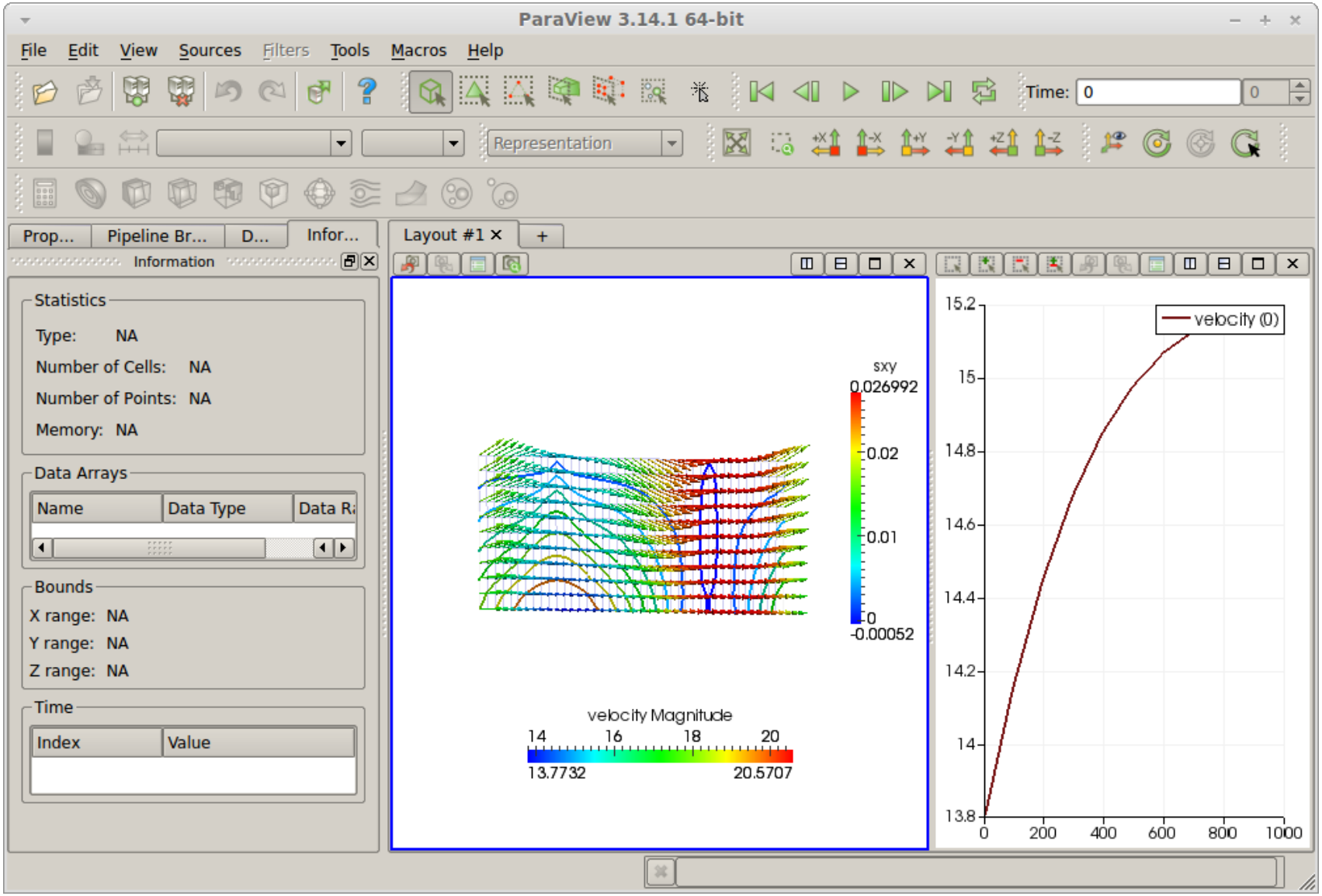
```
FUNCTION proc( Model, n, T ) RESULT(dens)
USE DefUtils
IMPLICIT None
TYPE(Model_t) :: Model
INTEGER :: n
REAL(KIND=dp) :: T, dens

    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...
Information
Statistics
Type: NA
Number of Cells: NA
Number of Points: NA
Memory: NA
Data Arrays
Name Data Type Data R
Bounds
X range: NA
Y range: NA
Z range: NA
Time
Index Value
Layout #1
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
```