Inverse methods implemented in Elmer/Ice (1)

Fabien Gillet-Chaulet, LGGE – Grenoble, gillet-chaulet@lgge.obs.ujf-grenoble.fr

- **Robin Inverse method** (Arthern and Gudmundsson, 2010)
  - Based on the computation of :
    - the « usual » Stokes problem (natural Neumann condition on the free surface)
    - the Dirichlet problem (observed surface velocities imposed as Dirichlet condition son the free surface)

- **Control Inverse method** (Mac Ayeal, 1993, Morlighem et al., 2010, Petra et al., 2012, ...)
  - Based on the computation of the **Adjoint state** (the **non linear stokes** problem is **self-adjoint** when equipped with the Newton linearisation (Petra et al., 2012))

- **Efficient minimisation of the cost function**
  - Minimisation is done using the M1QN3 library (Gilbert and Lemaréchal, 1989), based on a limited memory quasi-Newton algorithm (**L-BFGS method**)

- **Already successfully applied to infer the badly known basal friction field**
  - Jay-Allemand et al., 2011; Shäfer et al, 2012; Gillet-Chaulet et al, 2012

- **Solvers** for the inversion **of the basal friction now** under the **"elmerice" repository**
  - Solvers for the inversion of the ice viscosity will follow shortly with documentation in the wiki and test cases
  - Adjoint solvers can be extended for the inversion of Neumann or Dirichlet boundary

Inverse methods implemented in Elmer/Ice (1)

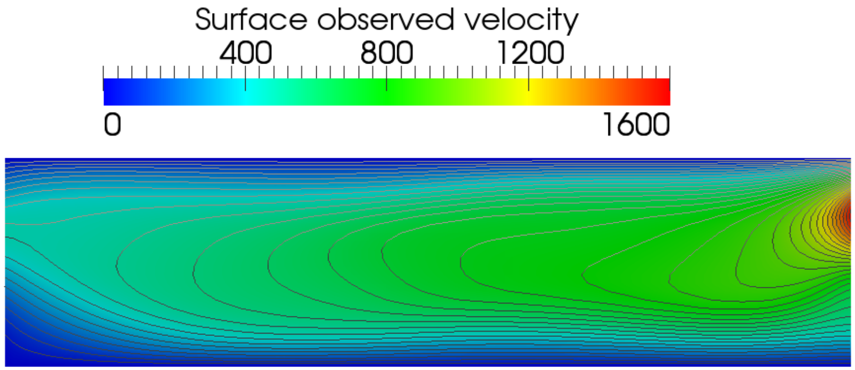
Fabien Gillet-Chaulet, LGGE – Grenoble, gillet-chaulet@lgge.obs.ujf-grenoble.fr

- **Robin Inverse method** (Arthern and Gudmundsson, 2010)
  - Based on the computation of :
    - the « usual » Stokes problem (natural Neumann condition on the free surface)
    - the Dirichlet problem (observed surface velocities imposed as Dirichlet condition son the free surface)
- **Control Inverse method** (Mac Ayeal, 1993, Morlighem et al., 2010, Petra et al., 2012, ...)
  - Based on the computation of the **Adjoint state** (the **non linear stokes** problem is **self-adjoint** when equipped with the Newton linearisation (Petra et al., 2012))
- **Efficient minimisation of the cost function**
  - Minimisation is done using the M1QN3 library (Gilbert and Lemaréchal, 1989), based on a limited memory quasi-Newton algorithm (**L-BFGS method**)
- **Already successfully applied to infer the badly known basal friction field**
  - Jay-Allemand et al., 2011; Shäfer et al, 2012; Gillet-Chaulet et al, 2012
- **Solvers** for the inversion **of the basal friction now** under the **“elmerice” repository**
  - Solvers for the inversion of the ice viscosity will follow shortly with documentation in the wiki and test cases
  - Adjoint solvers can be extended for the inversion of Neumann or Dirichlet boundary conditions

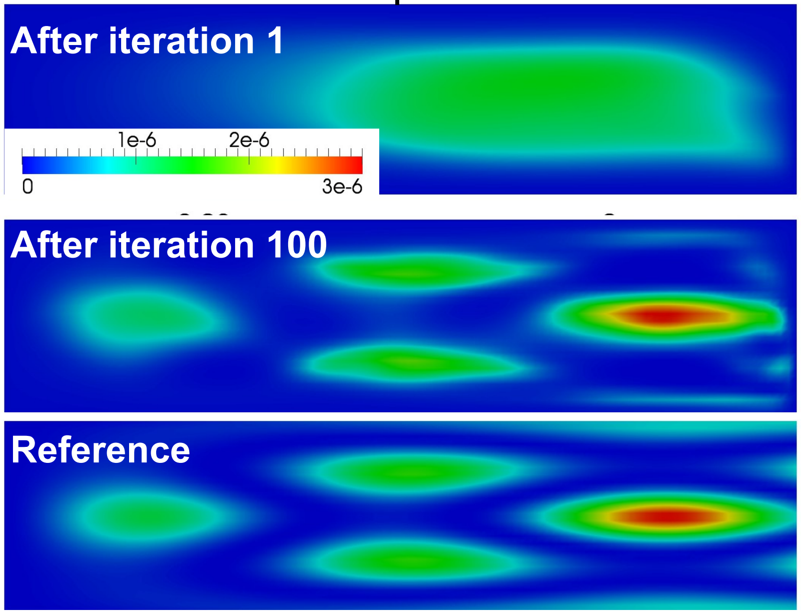
Inverse methods implemented in Elmer/Ice (2)

Fabien Gillet-Chaulet, LGGE – Grenoble, gillet-chaulet@lgge.obs.ujf-grenoble.fr

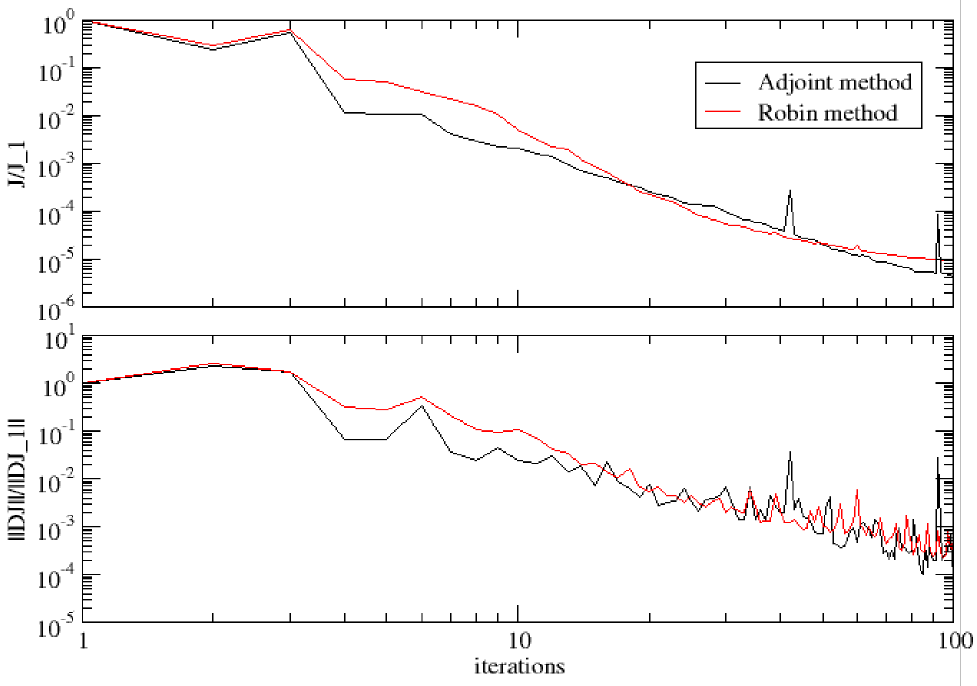
- **A test case** for the inversion of **the basal slip coefficient** is under:
  ELMER_HOME/elmerice/examples/InverseMethods
- **Twin experiments** based on **Mac Ayeal (1993) example**



Basal slip coefficient



Cost function and norm of the gradient as a function of the number of iterations



Scattered 2D Data Interpolator (1)

Fabien Gillet-Chaulet, LGGE – Grenoble, gillet-chaulet@lgge.obs.ujf-grenoble.fr

- **Interpolation of scattered 2D data** (e.g, ice thickness along flight lines, etc...) **onto the FE mesh**

- **Scattered data** are given under the form of **3-columns ASCII files** (x,y,value)

- **Natural Neighbours interpolation** or **cubic spline approximation**
  - Based on external c-librairies
    - nn (http://code.google.com/p/nn-c/)
    - csa (http://code.google.com/p/csa-c/)
  - The user is advised to **get familiar with these libraries**

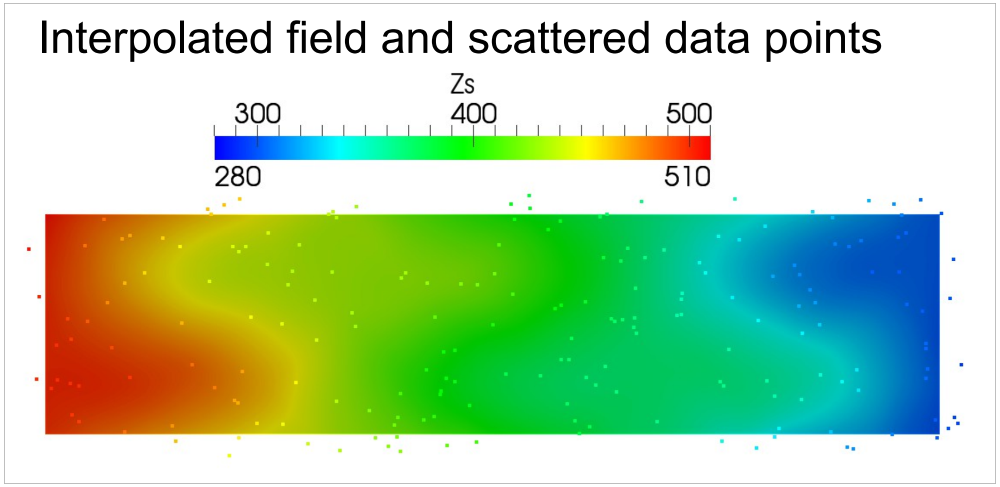
- **To compile the solver:**

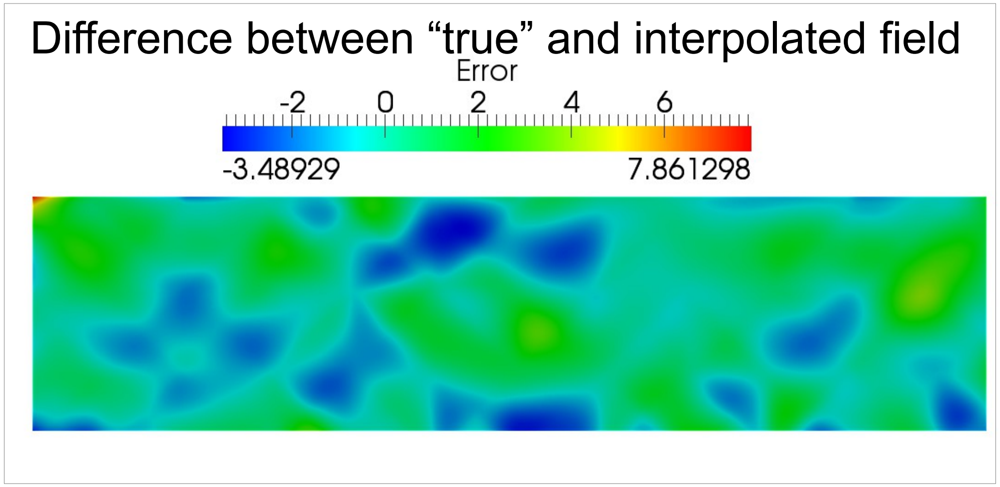
  1- Download/install these libraries on your favourite computer
  2- Edit/update the file “ELMER_HOME/elmerice/Solvers/MakefileScattered2D.inc”
  3- Compile the ElmerIceSolver library

Scattered 2D Data Interpolator (2)

Fabien Gillet-Chaulet, LGGE – Grenoble, gillet-chaulet@lgge.obs.ujf-grenoble.fr

- **A test case** is under:
  ELMER_HOME/elmerice/examples/Scattered2DDataInterpolator
  - “True” $Zs = 500 - 10^{-3} x + 20 (\sin(3 \pi x / L_x) \sin(2 \pi y / L_y))$
  - Generate 200 points at random locations
  - Interpolate on the FE mesh using the c libraries


Interpolated field and scattered data points


Difference between “true” and interpolated field

- **Future developments:**
  - Read NETCDF files
  - Use the ability of the csa library to use standard error