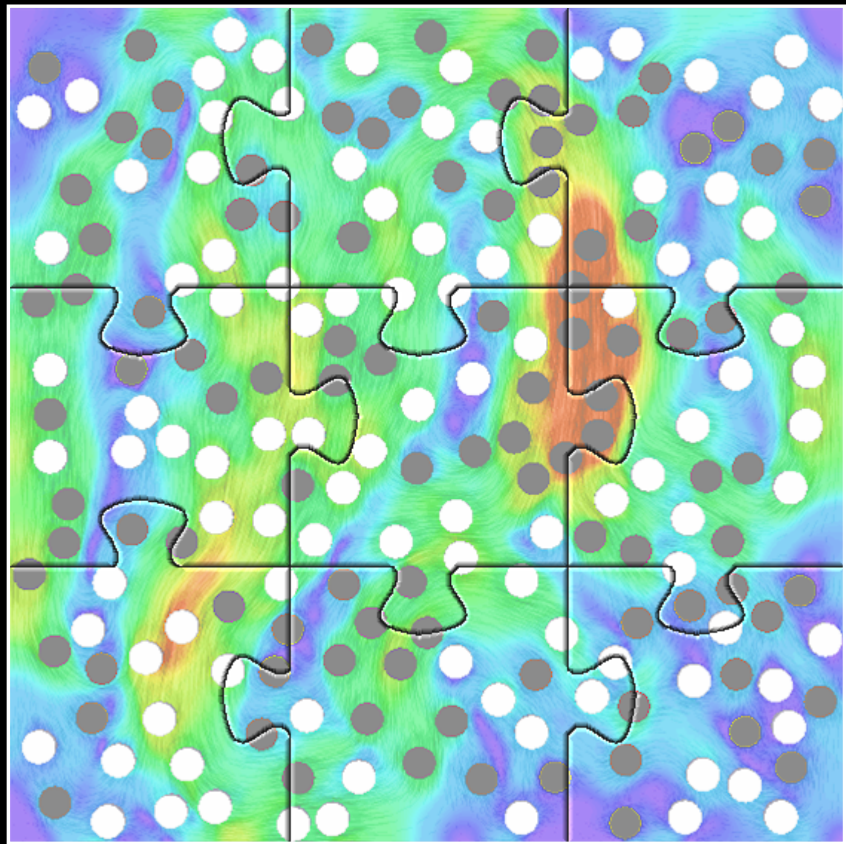
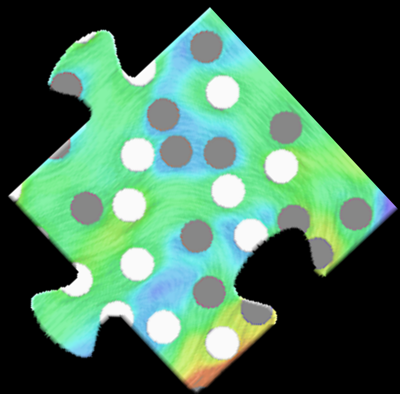
Unravelling the puzzle of numerical modeling



Marcin Dabrowski, Marcin Krotkiewski, Dani Schmid

PGP, Oslo

Unstructured FEM

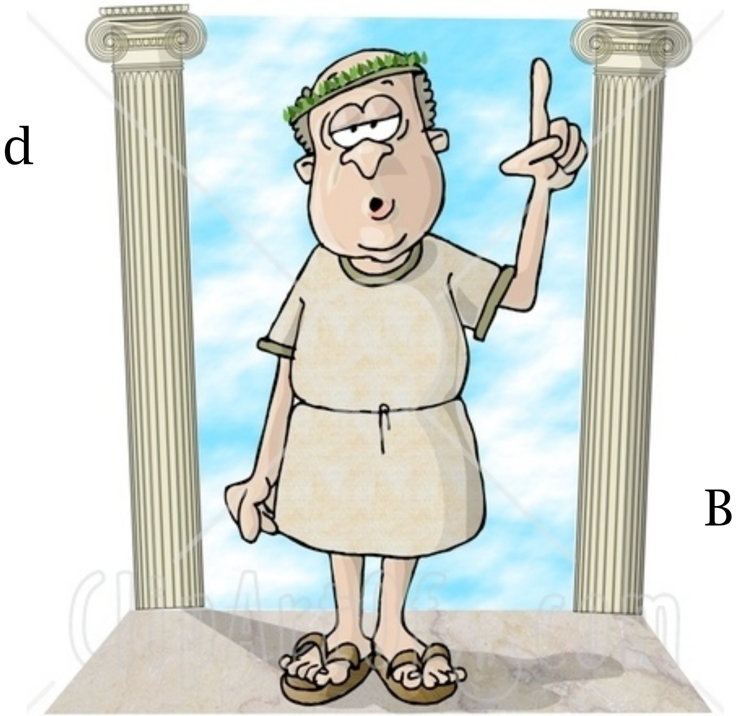


What method to use?

Finite Element Method

Finite Difference Method

Spectral Method



Finite Volume Method

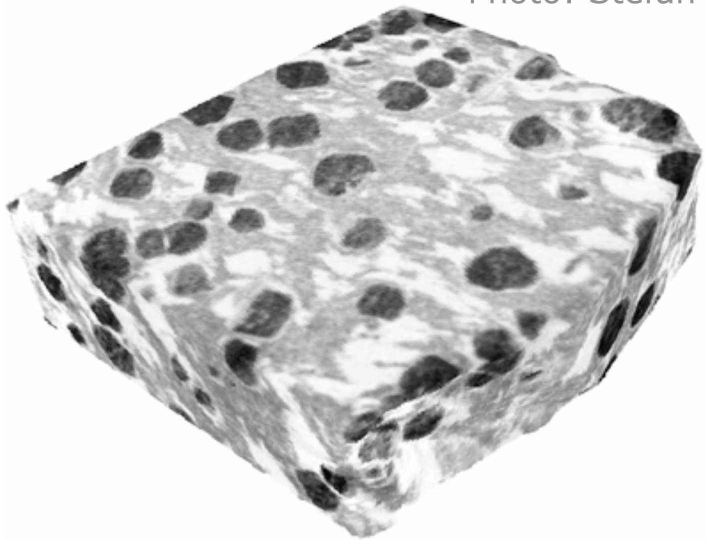
Boundary Element Method

Meshfree Method

Complex geometry – natural examples

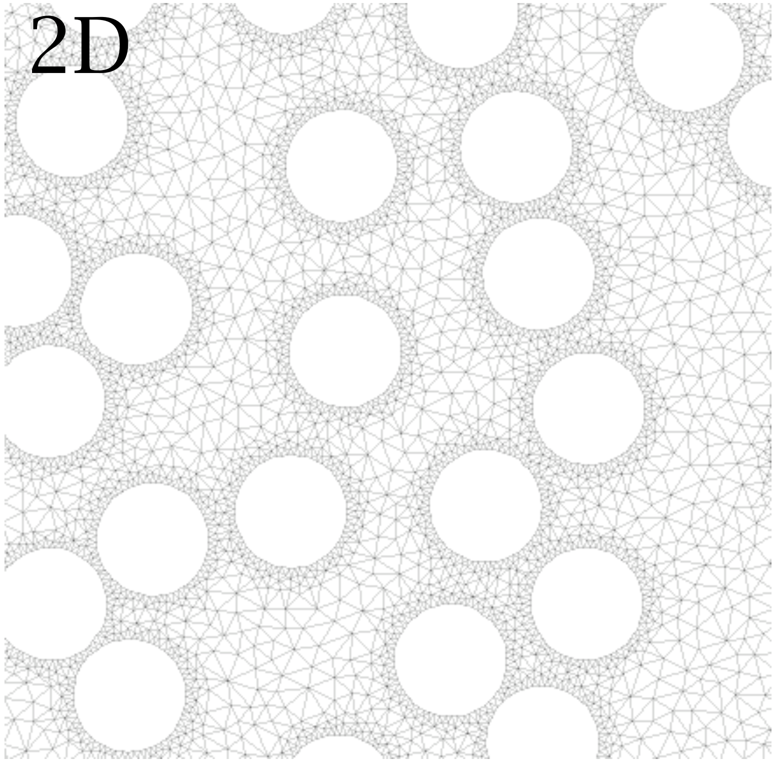
Photo: Stefan Schmalholz

Inclusions in Sandane shear zone (Norway)

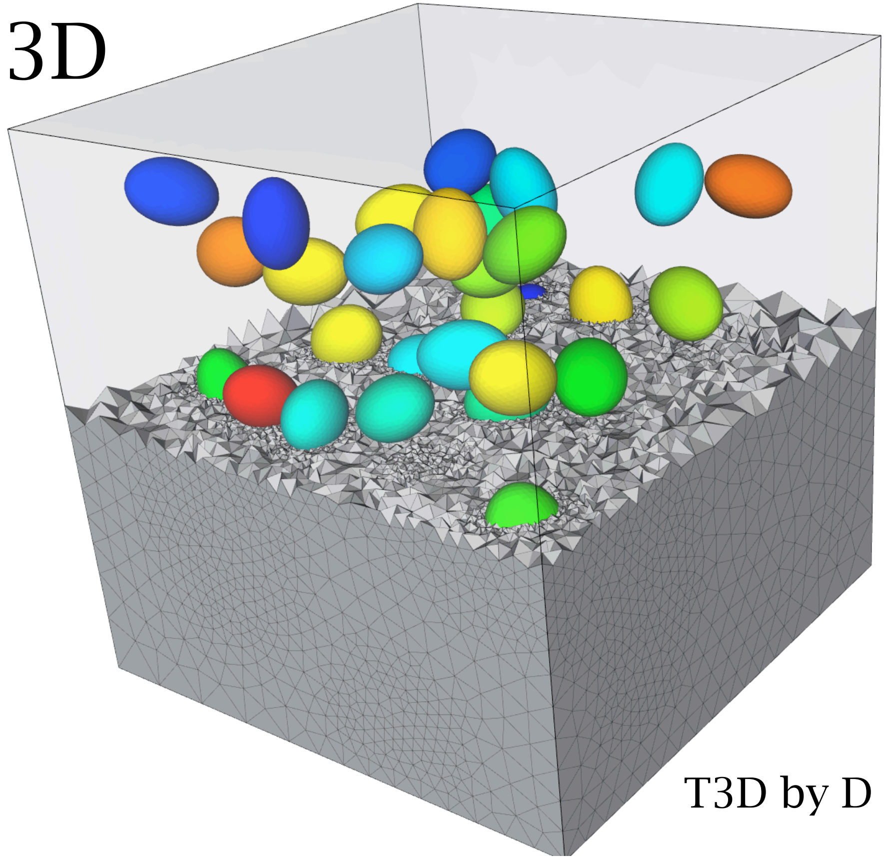


CT scan of migmatitic garnet amphibolite by Micheal Brown

Complex geometry – mesh generation



Triangle by JR Shewchuk



T3D by D Rypl

Unstructured finite element method (FEM)

Approximation space

$$T(u,v) = \sum_i T_i N_i(u,v)$$

Derivatives

$$\frac{\partial T(\vec{x})}{\partial x_i} = \sum_k T_k \frac{\partial N_k(\vec{u})}{\partial u_j} \frac{\partial u_j}{\partial x_i}$$

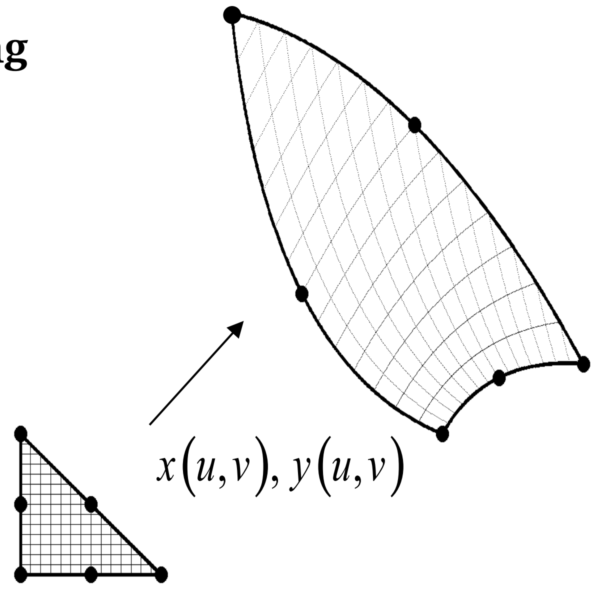
Weighted residual method

$$\int_\Omega N_i \left(\nabla \cdot k \nabla T + Q\right) d\vec{x} = 0$$

Integration by parts

$$\int_\Omega k \nabla N_i \cdot \nabla T d\vec{x} = \int_\Omega N_i Q d\vec{x} + \int_{\partial\Omega} N_i q_n ds$$

Mapping



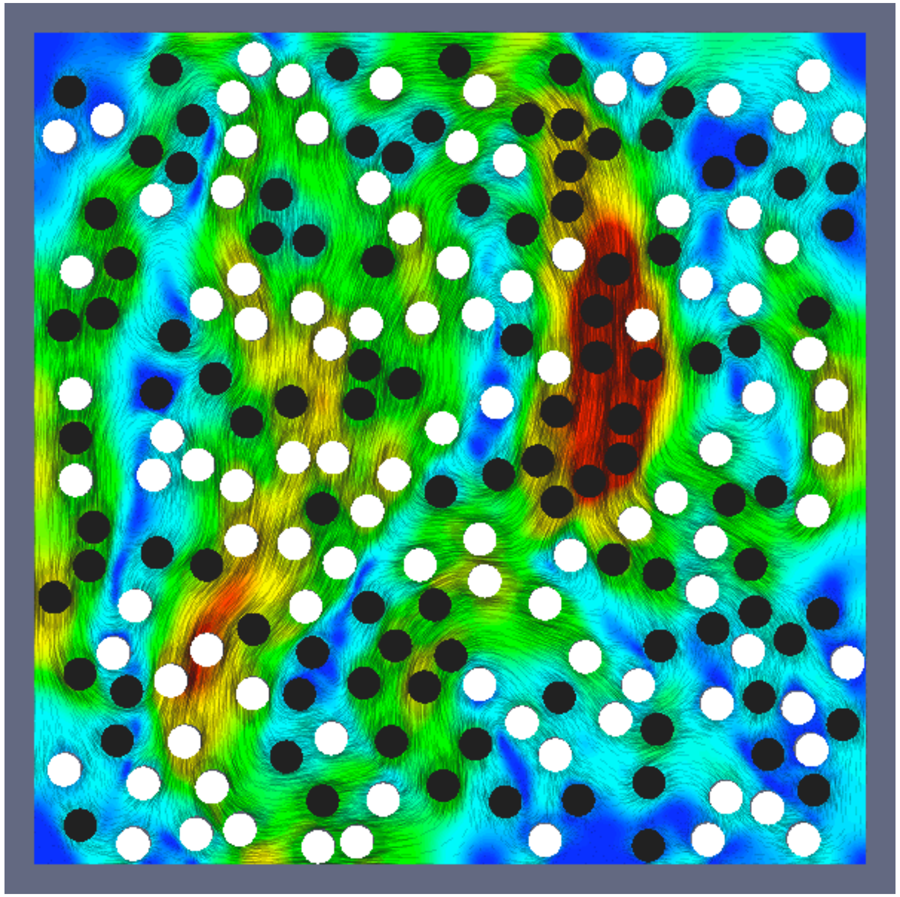
Is it easy? YES!

Unstructured FEM – matrix computation

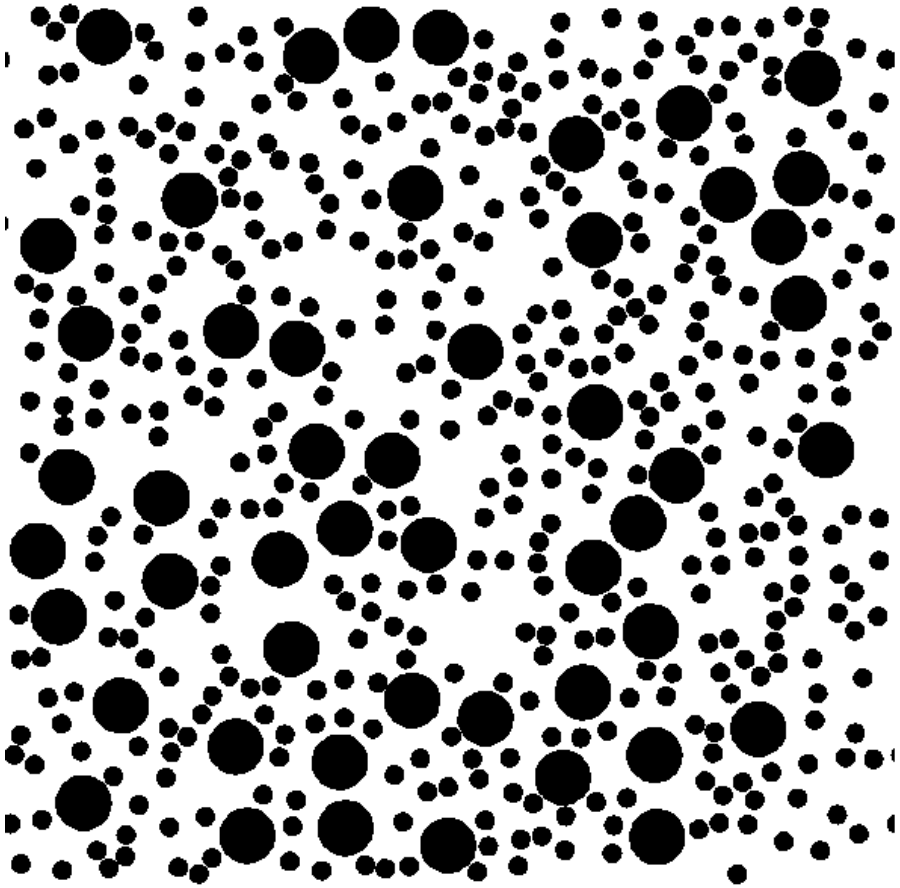
```
for iel = 1:nel
    ECOORD_X = GCOORD(:,ELEM2NODE(:,iel));
    ED = D(Phases(iel));
    K_elem(:) = 0;
    for ip=1:nip
        dNdui = dNdu{ip};
        J = ECOORD_X*dNdui;
        detJ = det(J);
        invJ = inv(J);
        dNdX = dNdui*invJ;
        K_elem = K_elem +  IP_w(ip)*detJ*ED*(dNdX*dNdX');
    end
    K_all(:,iel) = K_elem(:);
end
```

Just Do It

Large deformation – unstructured FEM

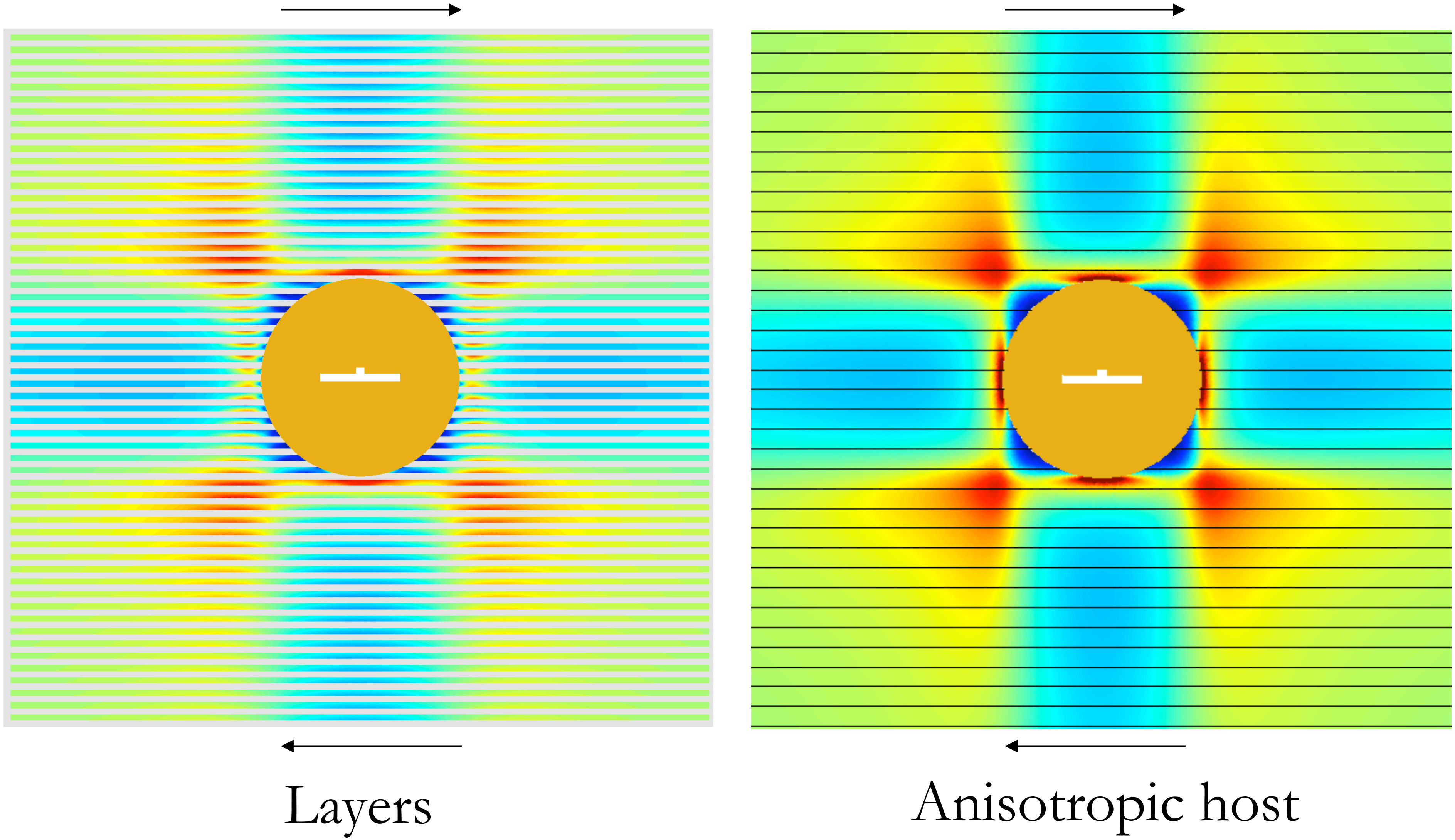


Particles settling under gravity

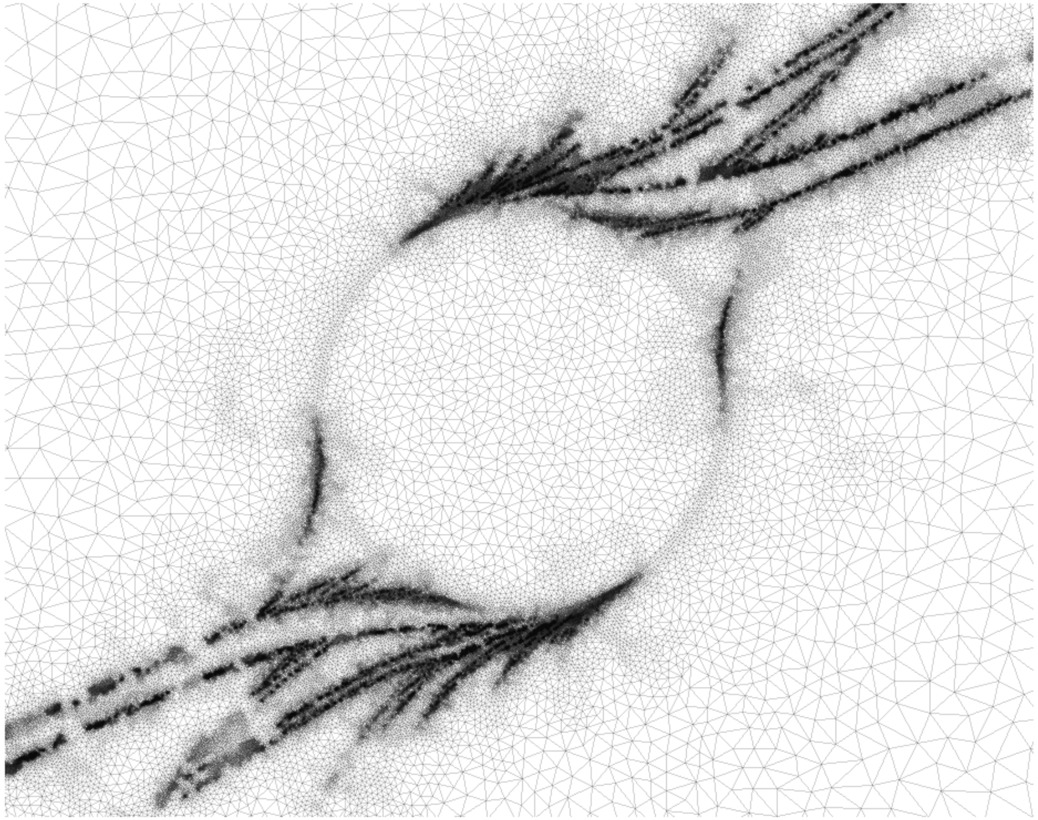


Rigid inclusions in simple shear

Adaptive unstructured FEM

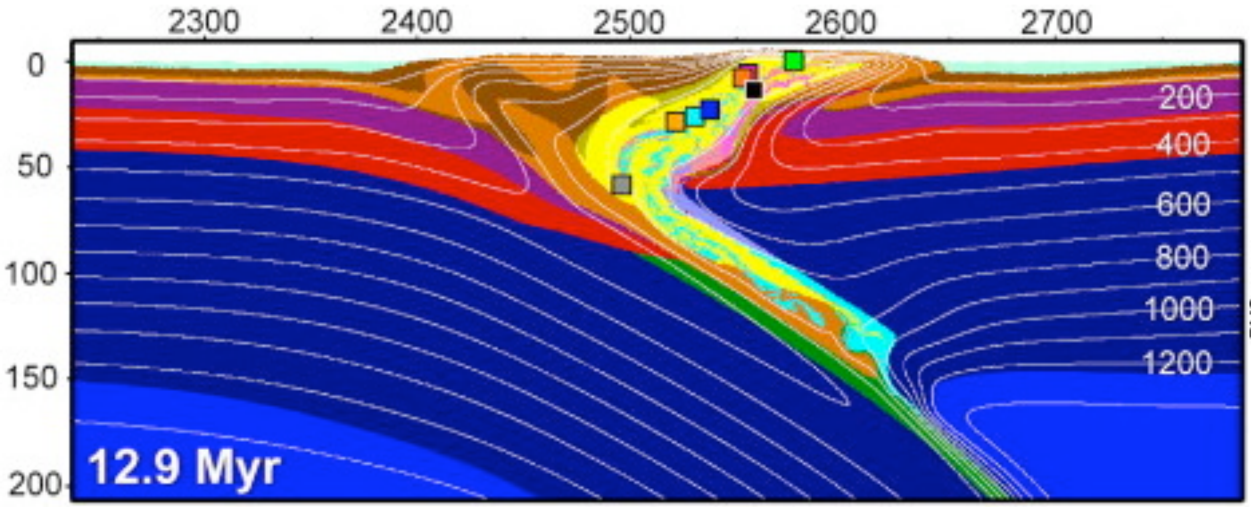


Adaptive unstructured FEM - accuracy

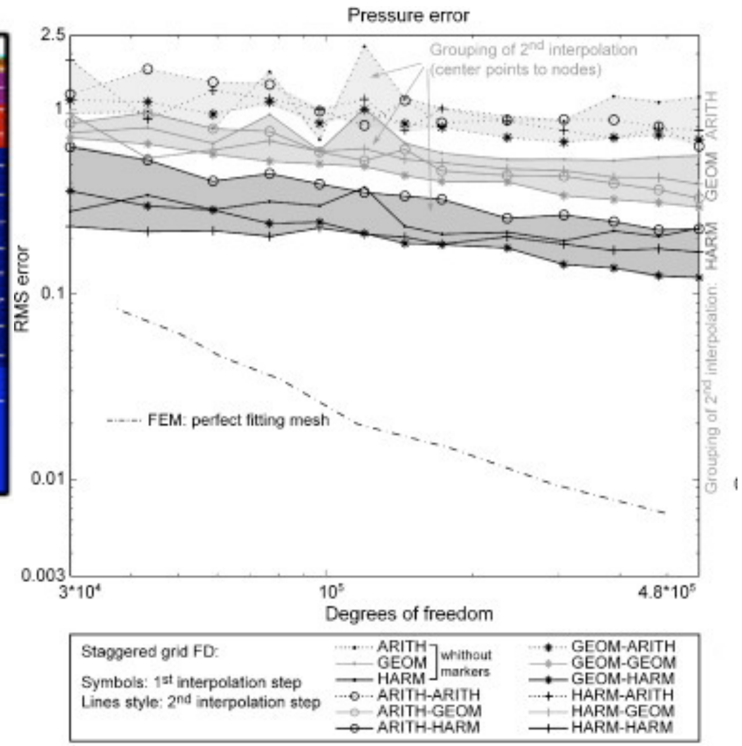


mesh adaptation:
a posteriori error analysis
& anisotropy structure

Complexity on geodynamic scale



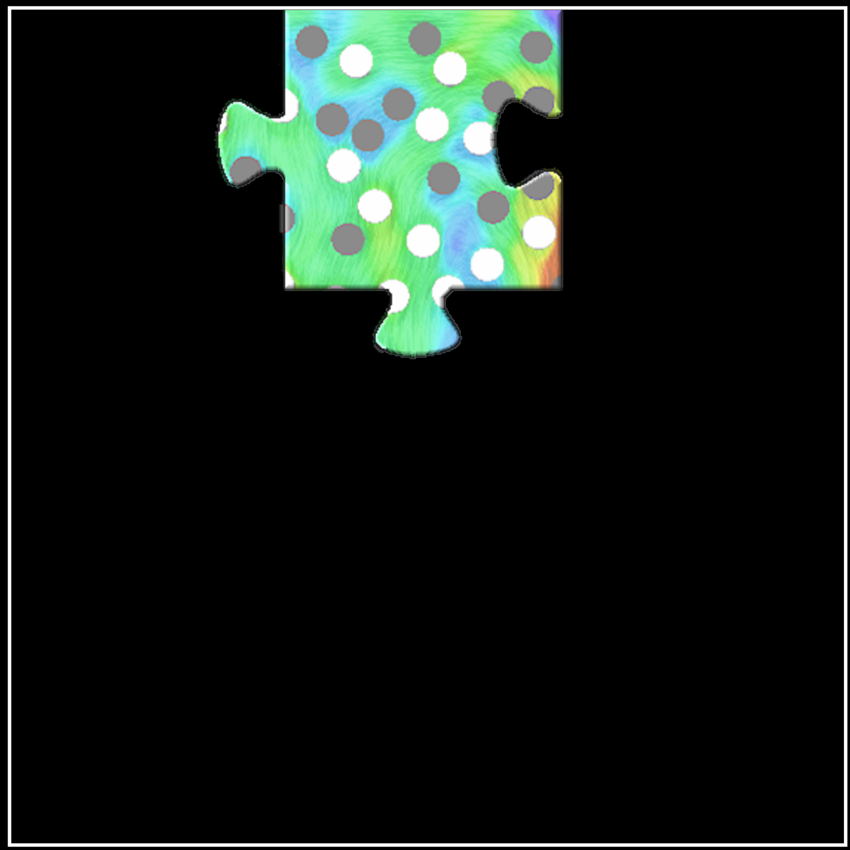
Gerya et al., Lithos, 2008



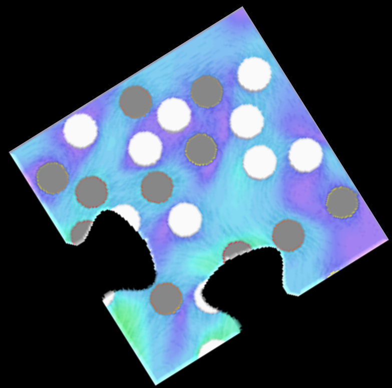
What about the accuracy?
Could unstructured FEM method become a standard numerical tool for 2D and 3D geodynamic simulations?

Deubelbeiss & Kaus, PEPI, 2008

Unstructured FEM delivers accurate results
Matrix computation can be efficient in 2D & 3D



Direct solvers



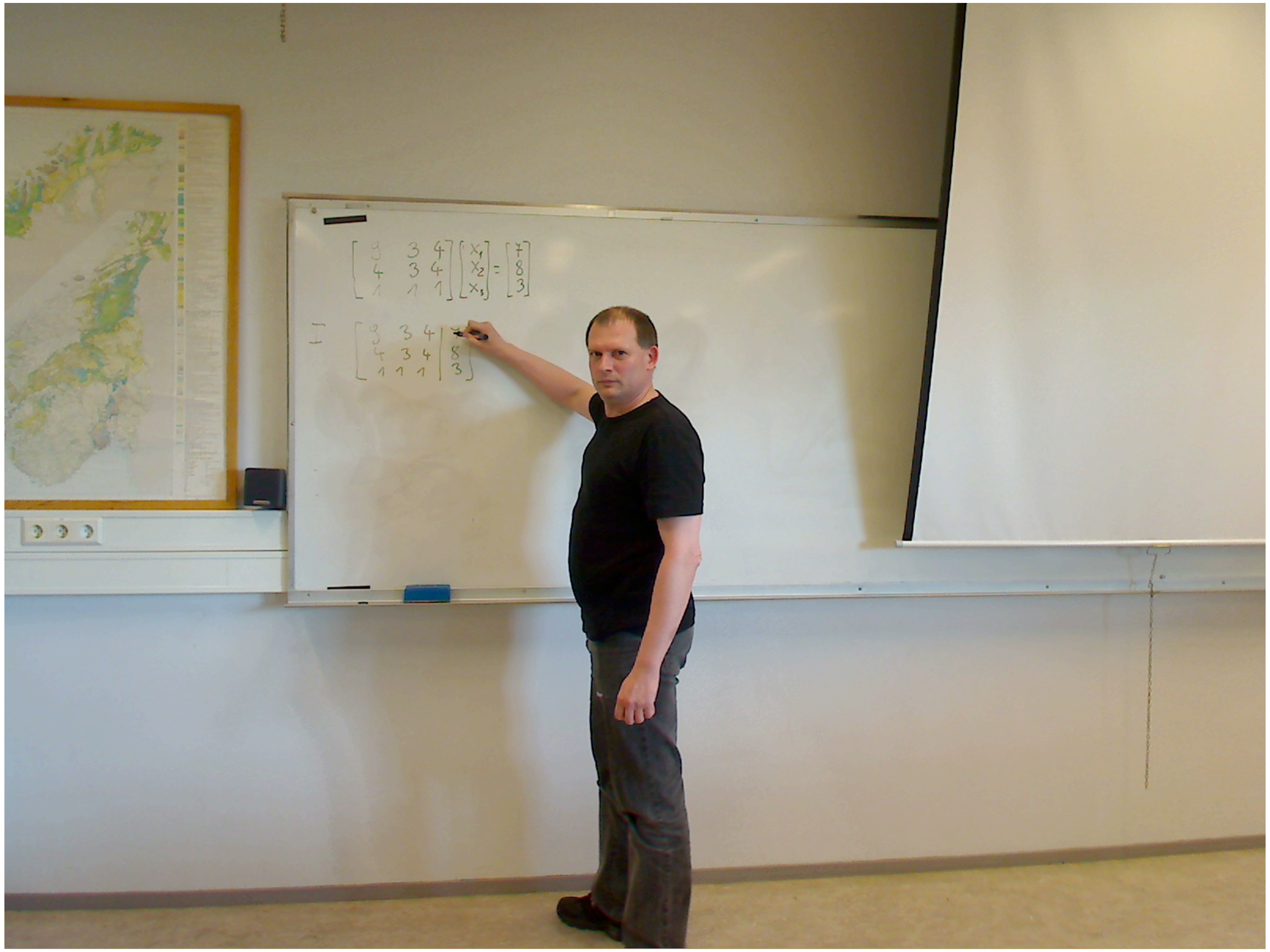
Gaussian elimination

Sergei Medvedev performing Gaussian elimination at PGP

Carl Friedrich Gauss

0 sec

Gaussian elimination



Carl Friedrich Gauss

35 sec

Sergei Medvedev performing Gaussian elimination at PGP

Gaussian elimination

Sergei Medvedev performing Gaussian elimination at PGP

Carl Friedrich Gauss

68 sec

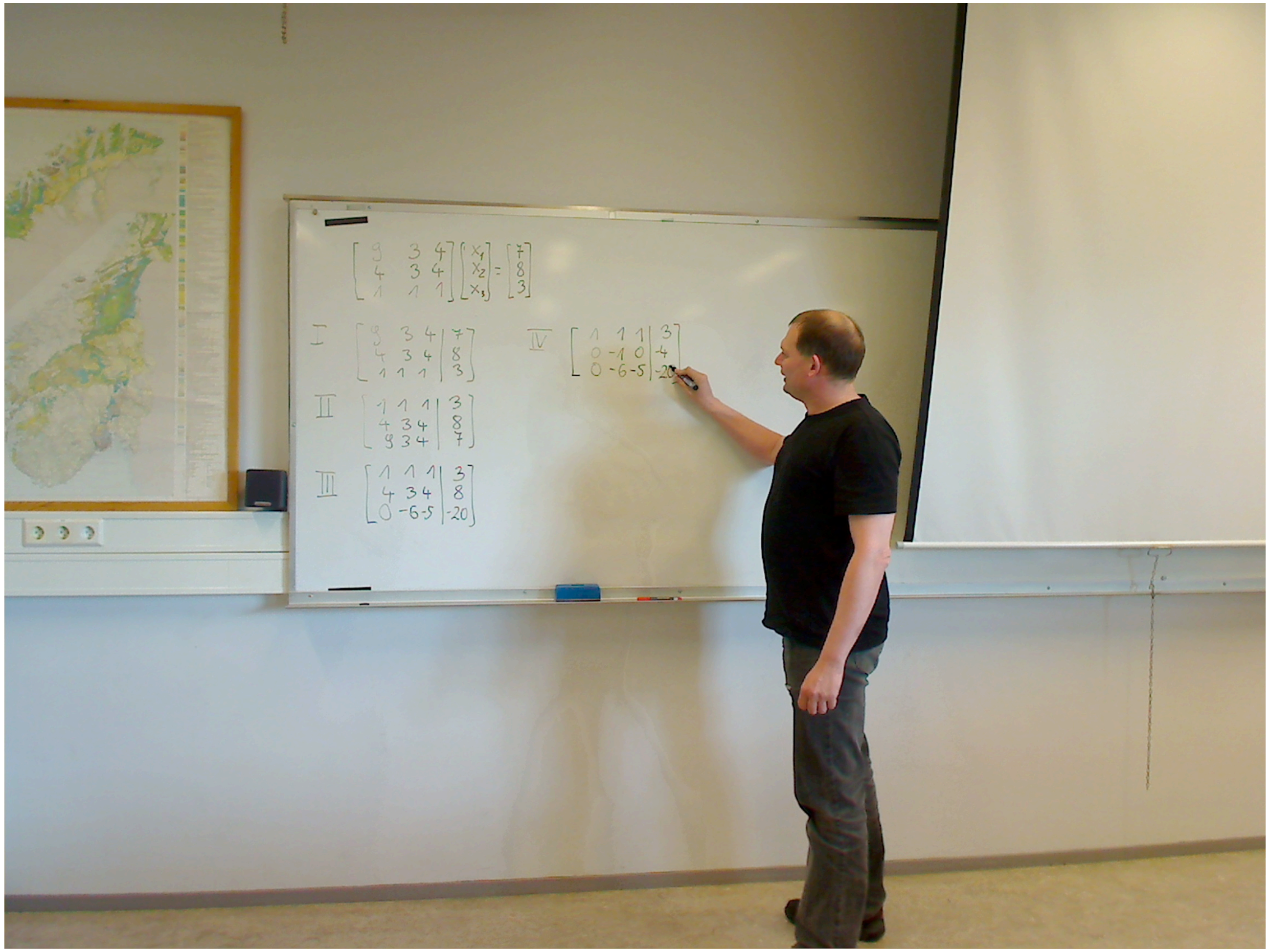
Gaussian elimination

Sergei Medvedev performing Gaussian elimination at PGP

Carl Friedrich Gauss

135 sec

Gaussian elimination

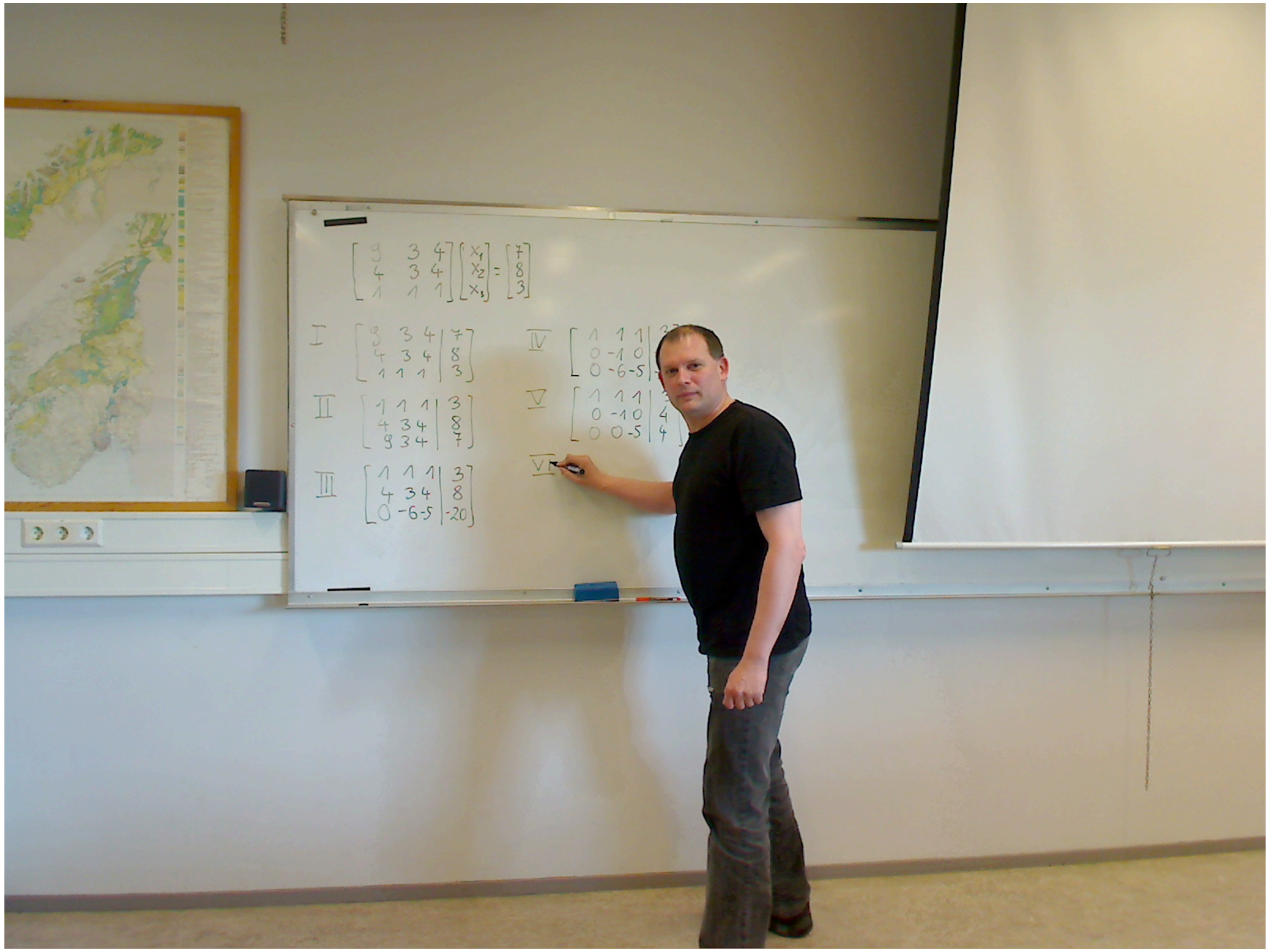


Sergei Medvedev performing Gaussian elimination at PGP

Carl Friedrich Gauss

200 sec

Gaussian elimination

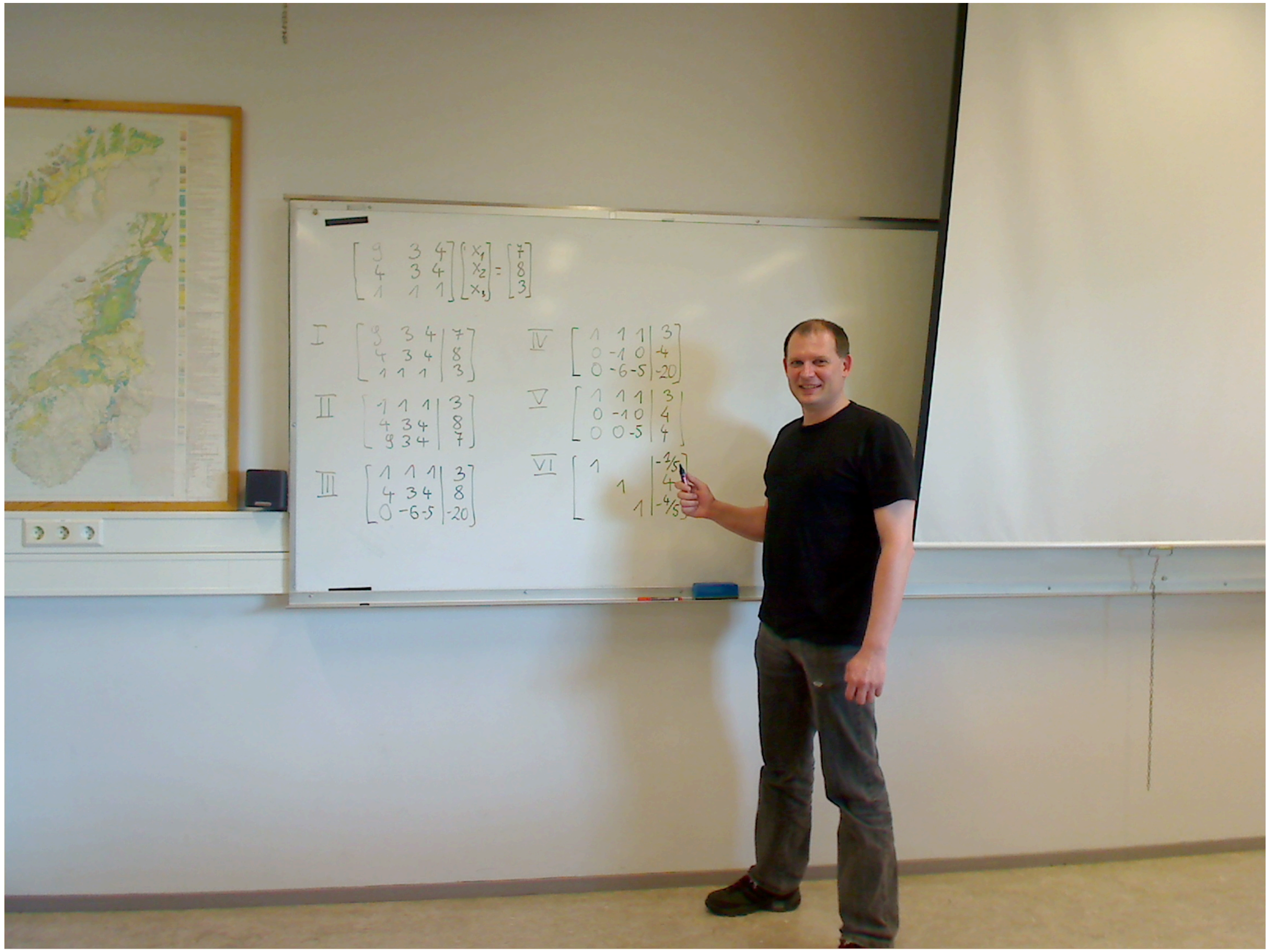


Sergei Medvedev performing Gaussian elimination at PGP

Carl Friedrich Gauss

250 sec

Gaussian elimination



Sergei Medvedev performing Gaussian elimination at PGP

Carl Friedrich Gauss

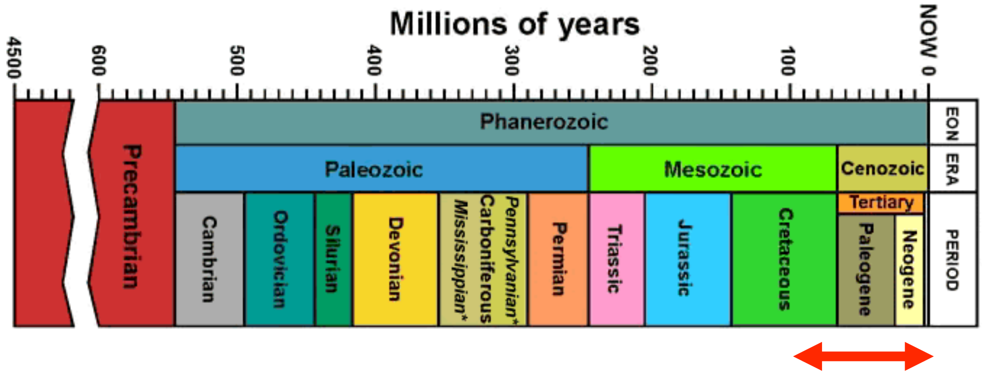
310 sec

Dense matrix format

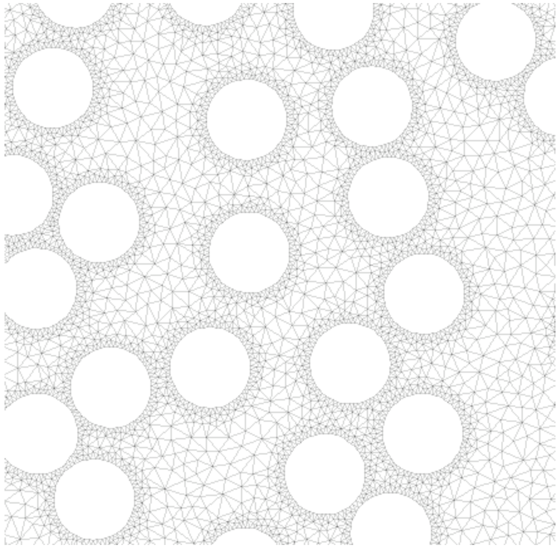
Storage $\sim n^2$

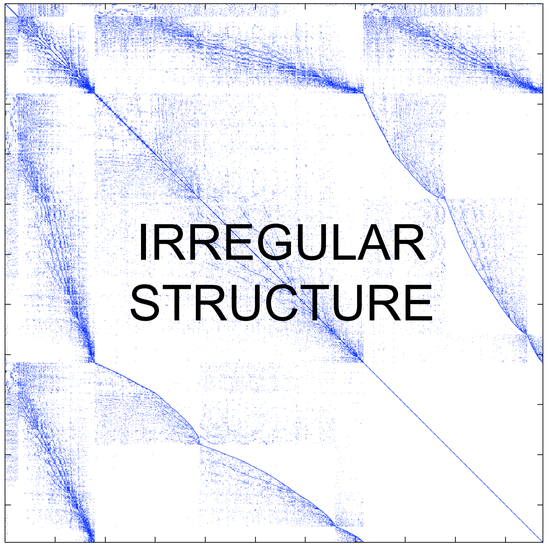
Operation count $\sim n^3$

Efficiency ~4 GFlops/CPU
(~0.1 Flops/Sergei)



100 Mdofs: PBytes storage
10^15 s ~ 100 Ma





Sparse matrix format

Matrix format Ti, Tj, Tx
(Ai, Aj, Ax)

Matrix storage $\sim n$

1 Mdofs(2D): 100 MB

100 Mdofs(3D): 50 GB

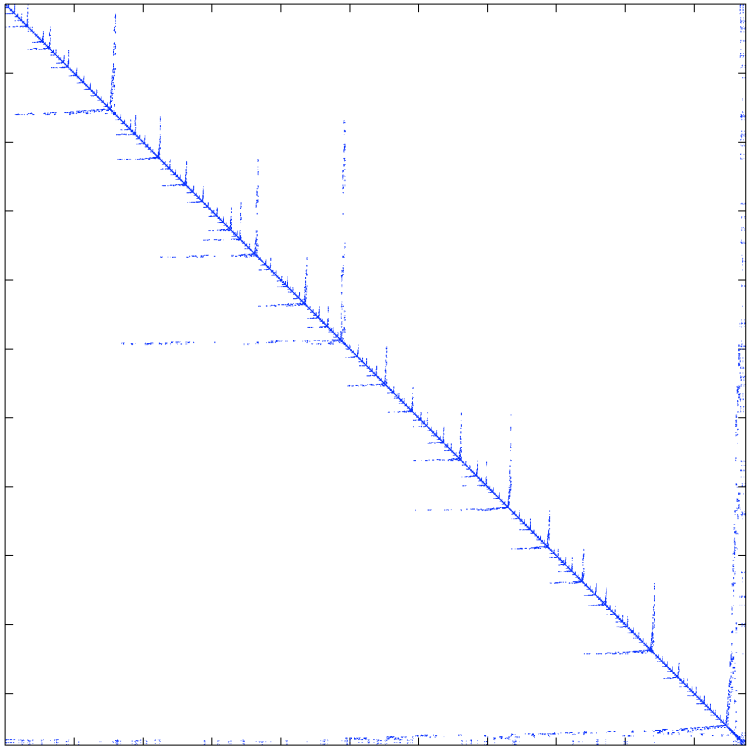
LU decomposition

$$Ax = b, \quad A = LU, \quad x = U^{-1}L^{-1}b$$

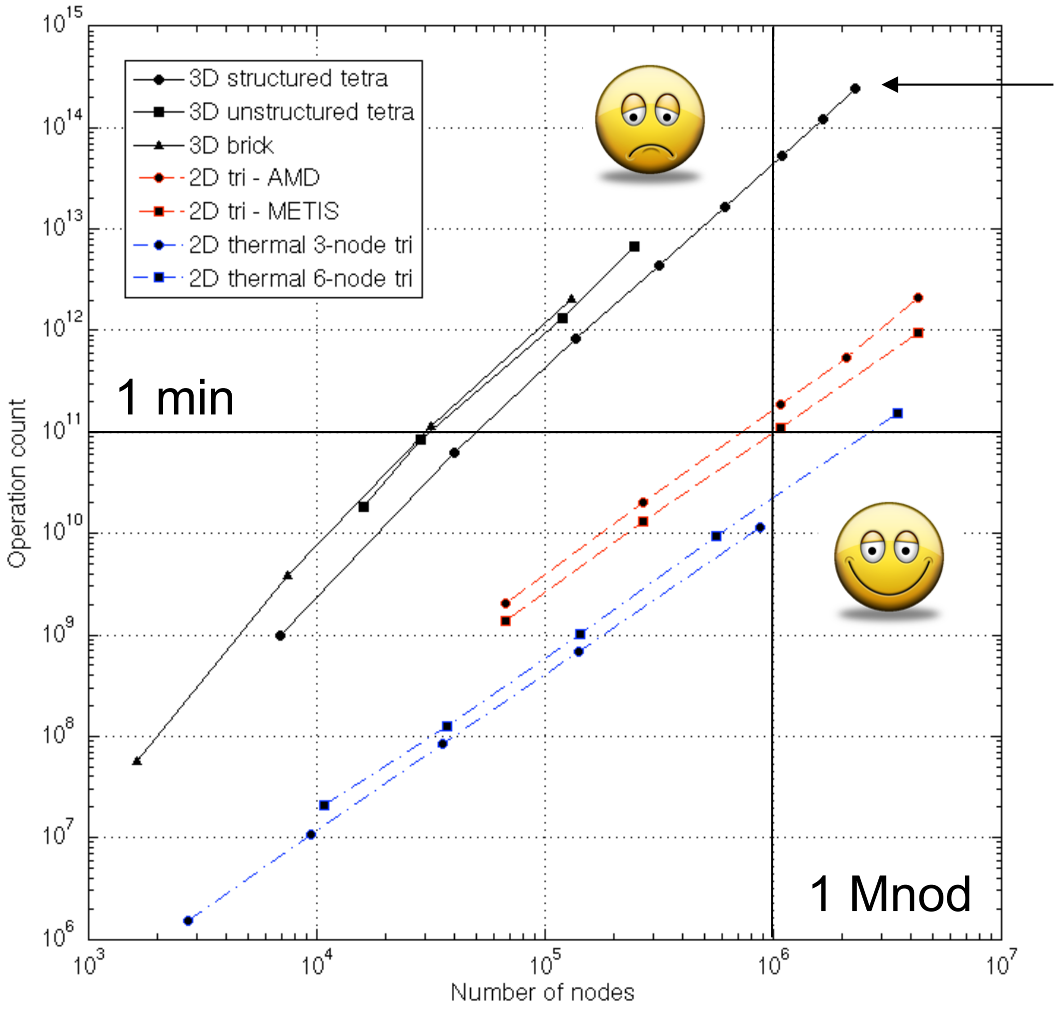
Cholesky factorization (U=L')

- Positive definite systems (thermal, elastic, bc!)
- Symbolic factorization (fill-in, op count)
- Reordering A(perm,perm)
- Factor storage (super-linear)
  - **3D**: nnz_L ~ nnod^1.4 (**40GB@1Mnod**)
  - **2D**: nnz_L ~ nnod^1.1 (**1.3GB@1Mnod**)

Fill-in reducing reordering



Operation count: scaling in 2D and 3D

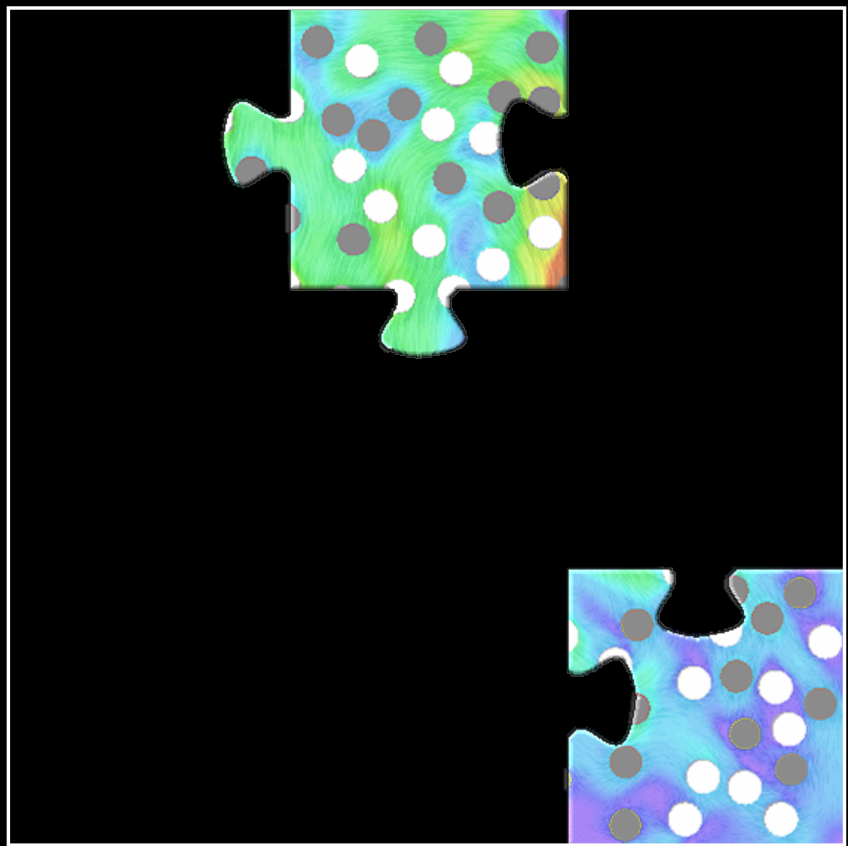


1.8 M nodes - 11216 sec [3h]
13 GFlops

Scaling
2D: op_count ~ nnod^1.5
3D: op_count ~ nnod^2

Efficiency
BLAS3 - 2 GFlops/CPU
Parallelization - difficult

Sparse direct solvers are great ...
for positive definite problems in 2D



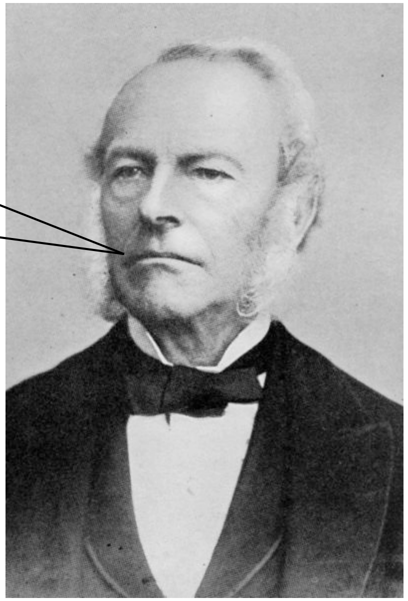
The Stokes problem

Incompressible Stokes problem

$$\nabla \cdot \mu \left(\nabla u + \nabla u^T \right) - \nabla p = f$$
$$\nabla \cdot u \qquad = 0$$

... and after FEM discretization

$$\begin{pmatrix} A & B^T \\ B & 0 \end{pmatrix} \begin{pmatrix} u \\ p \end{pmatrix} = \begin{pmatrix} f \\ 0 \end{pmatrix}$$



Sir George Gabriel Stokes

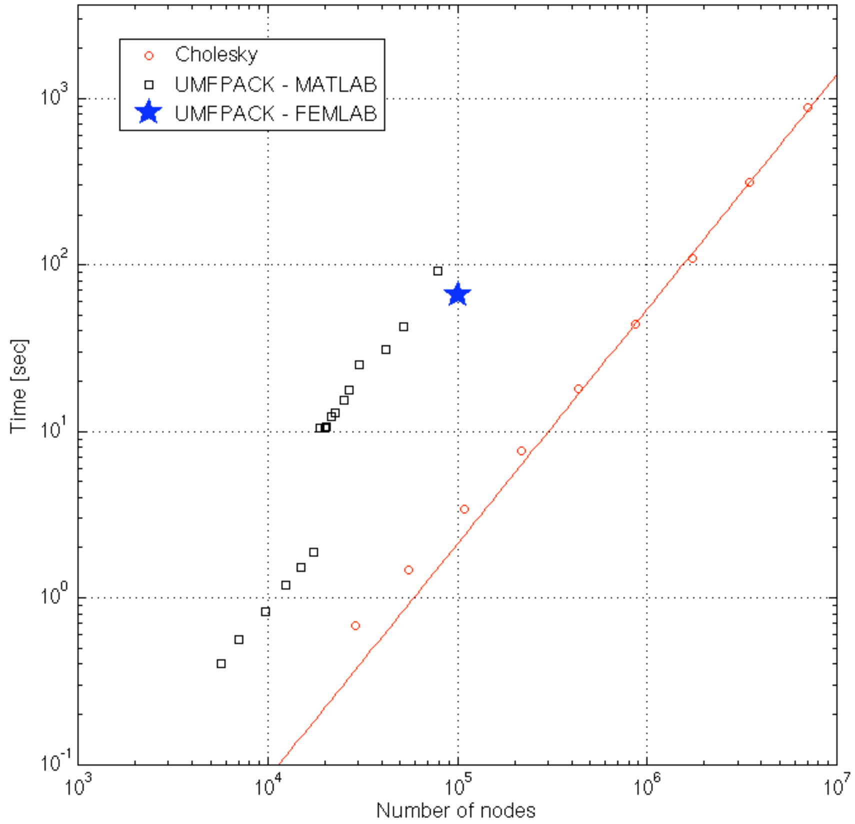
Indefinite systems – direct solvers

Discrete Stokes equations – indefinite!

$$\begin{pmatrix} A & B^T \\ B & 0 \end{pmatrix} \begin{pmatrix} u \\ p \end{pmatrix} = \begin{pmatrix} f \\ 0 \end{pmatrix}$$

Possible solvers:

1) unsymmetric sparse direct solvers
2) restore positive-definiteness
3) minres, gmres



Penalty method

$$\begin{pmatrix} A & B^T \\ B & -M/\kappa \end{pmatrix}\begin{pmatrix} u \\ p \end{pmatrix} = \begin{pmatrix} f \\ 0 \end{pmatrix}$$

Eliminate pressure ...

$$p = \kappa M^{-1} B u$$

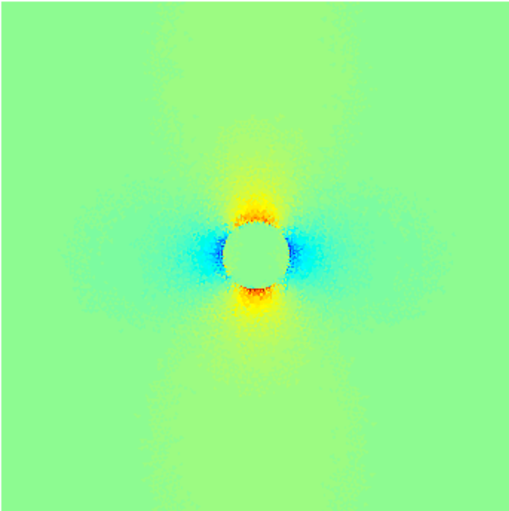
Velocity Schur complement

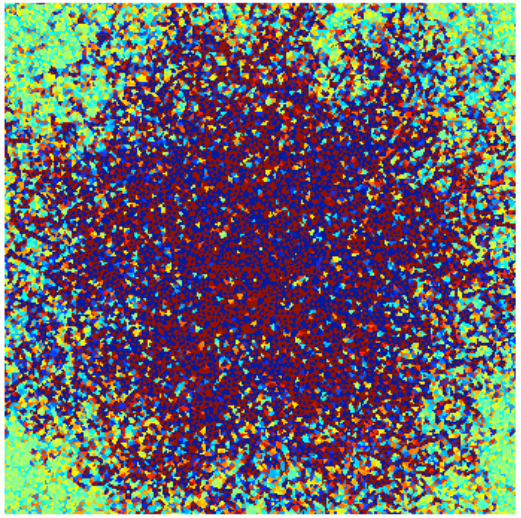
$$\left(A + \kappa B^T M^{-1} B\right) u = f$$

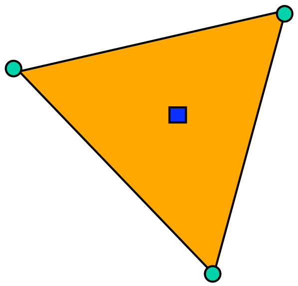
positive-definite system

Inclusion benchmark in 2D

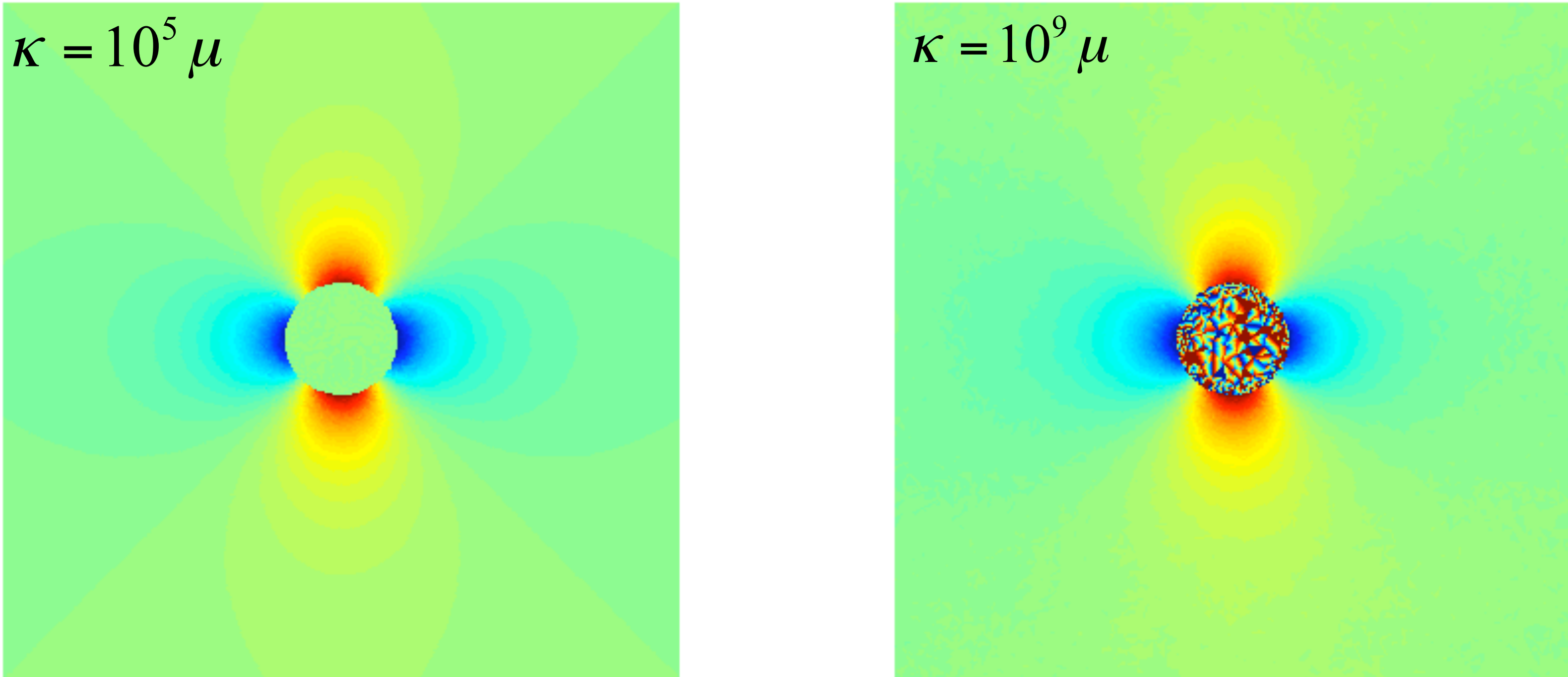
$\kappa = 10^1 \mu$ $\kappa = 10^3 \mu$



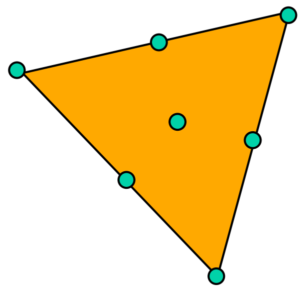




Linear velocity
Constant pressure



mixed u/p formulation: Crouzeix-Raviart 7-node triangle



Pressure Schur complement

Eliminate velocity dofs

$$\begin{pmatrix} A & B^T \\ B & 0 \end{pmatrix}\begin{pmatrix} u \\ p \end{pmatrix} = \begin{pmatrix} f \\ 0 \end{pmatrix} \rightarrow u = A^{-1}f - A^{-1}B^T p \rightarrow BA^{-1}B^T p = BA^{-1}f$$

Pressure Schur complement S (positive-definite)

$$Sp = b$$

$$S = BA^{-1}B^T, \quad A = LL^T, \quad b = BA^{-1}f$$

Matrix S cannot be formed explicitly (prohibitively costly)

$$y = Sx \qquad \begin{aligned} y_1 &= B^T x \\ y_2 &= L^{-T}\left(L^{-1}y_1\right) \\ y &= By_2 \end{aligned}$$

Augmented Lagrangian

$$\begin{pmatrix} A & B^T \\ B & 0 \end{pmatrix}\begin{pmatrix} u \\ p \end{pmatrix} = \begin{pmatrix} f \\ 0 \end{pmatrix} \Leftrightarrow \begin{pmatrix} A + \kappa B^T M^{-1} B & B^T \\ B & 0 \end{pmatrix}\begin{pmatrix} u \\ p \end{pmatrix} = \begin{pmatrix} f \\ 0 \end{pmatrix}$$

MILlion A MINute (MILAMIN)

Richardson iterations

$$Ax = b, \qquad x_{i+1} = x_i + \lambda\left(b - Ax_i\right)$$

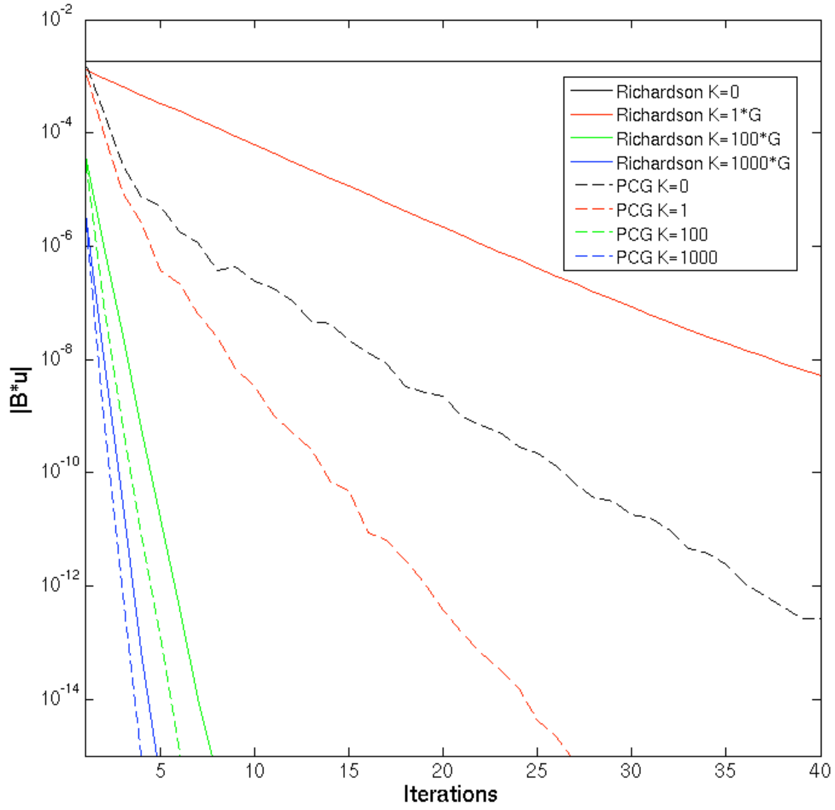
Iteration scheme

$$p_{i+1} = p_i + \left(M/\kappa\right)^{-1}\left(b - S(\kappa)p_i\right)$$

Implementation

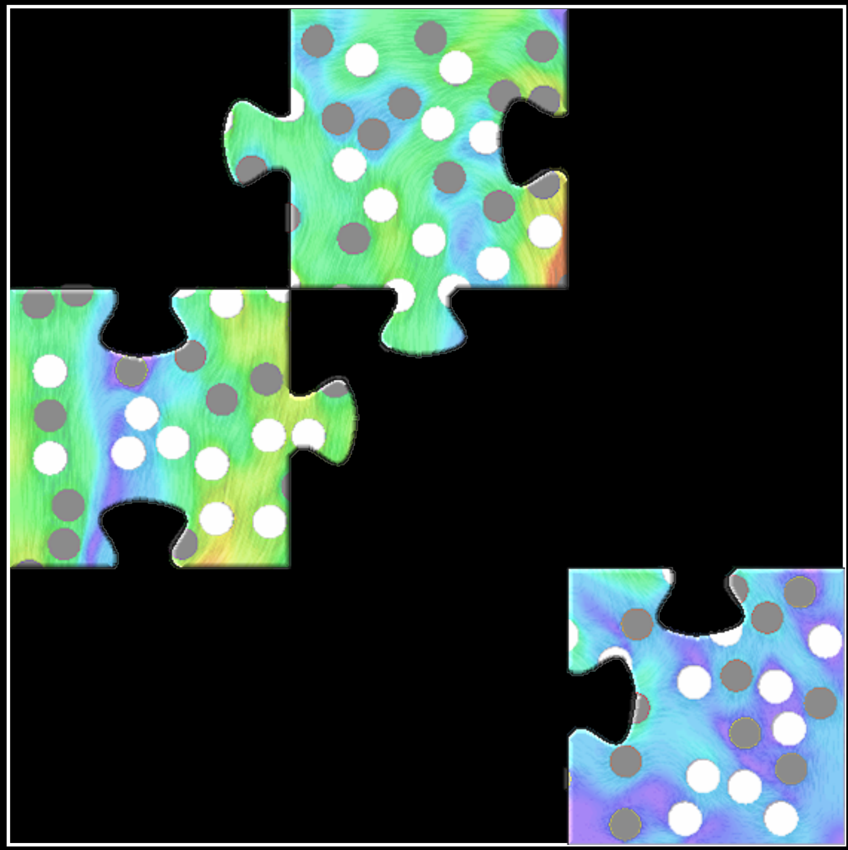
$$u_i = A(\kappa)^{-1}\left(f - B^T p_i\right)$$

$$p_{i+1} = p_i + \kappa M^{-1} B u_i$$

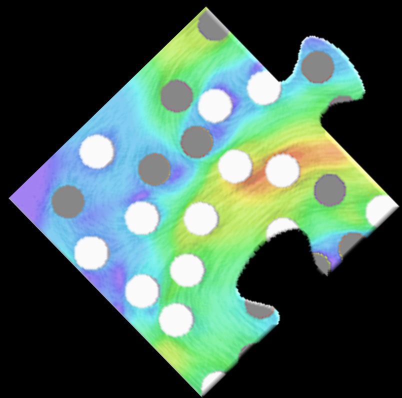


Dabrowski, M., M. Krotkiewski, and D. W. Schmid (2008), **MILAMIN: MATLAB-based finite element method solver for large problems**, G^3., 9

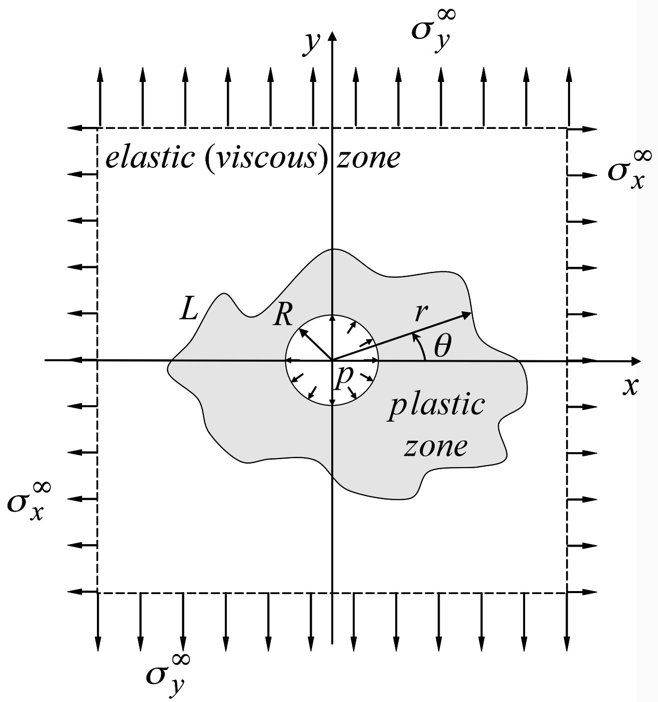
Stokes problem in 2D - no headaches and no tears

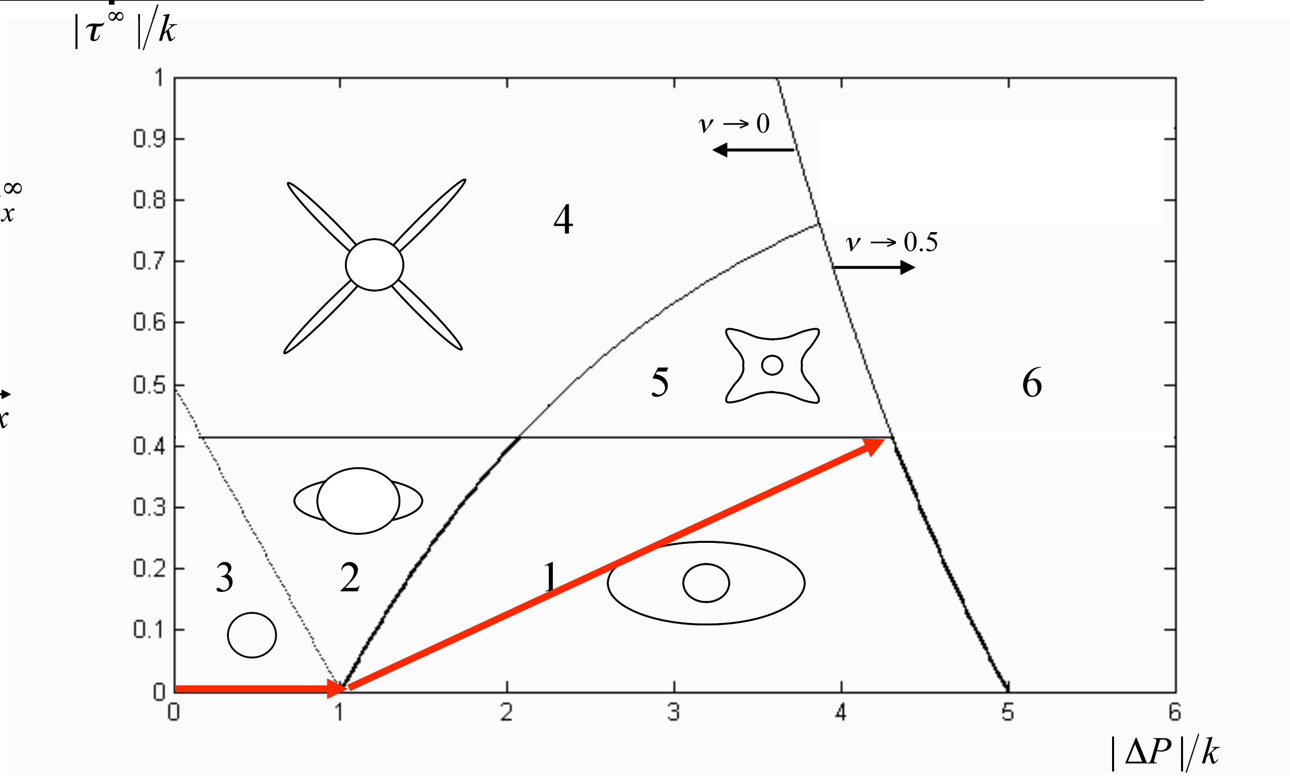


Non-linear iterations



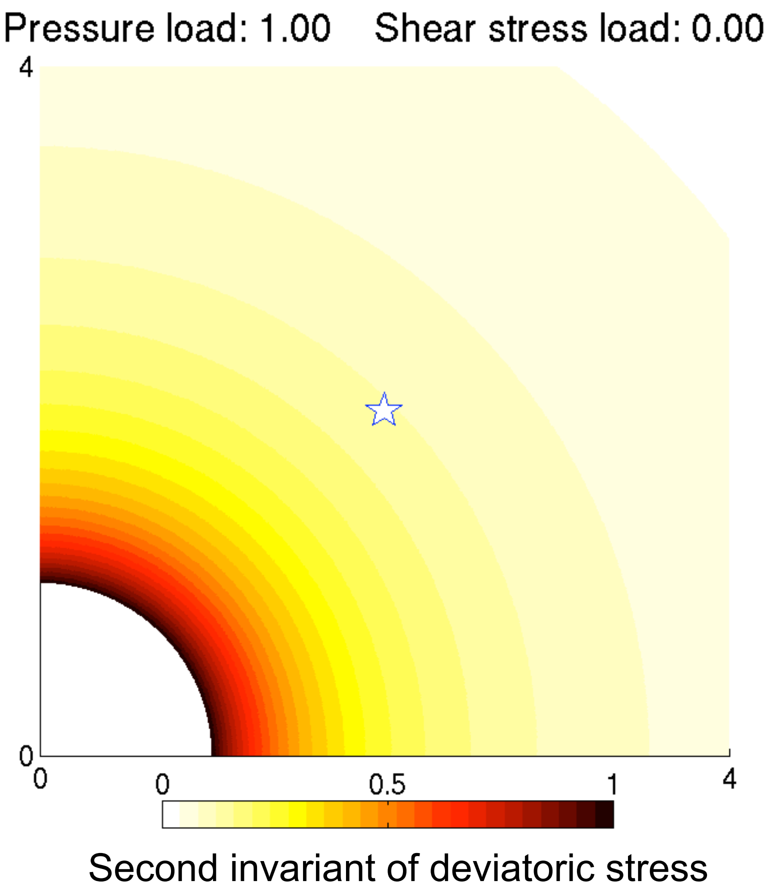
Elasto-plastic benchmark – Galin's solution

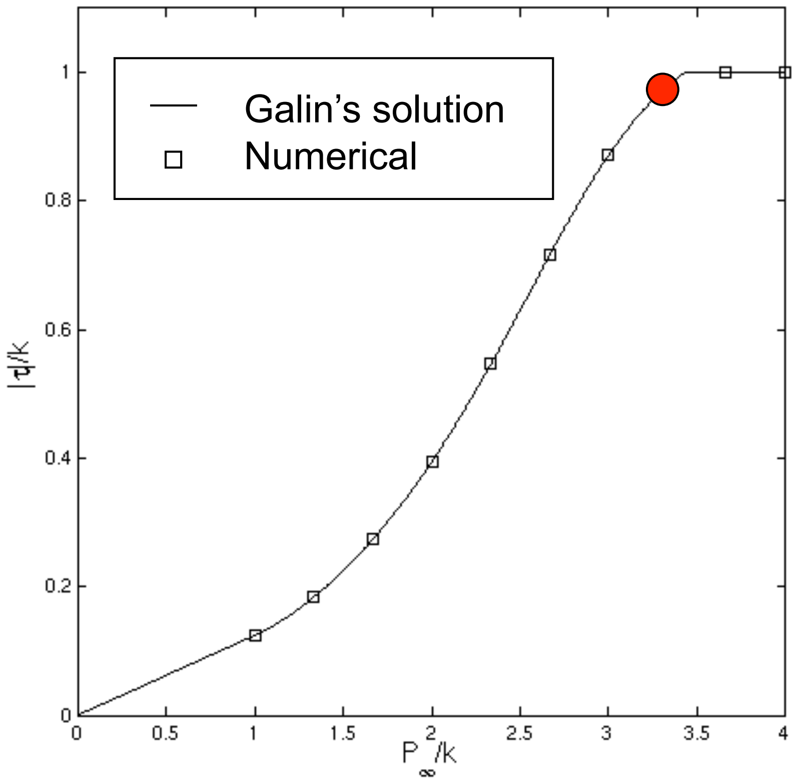




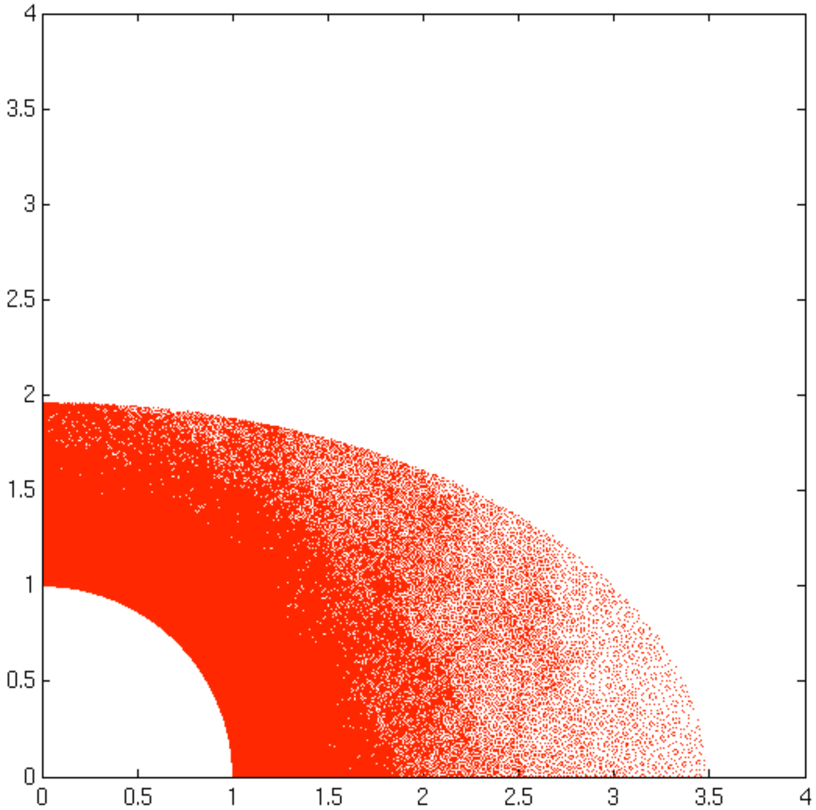
Yarushina V.M., Dabrowski M., Podladchikov Y.Y., An analytical benchmark with combined pressure and shear loading for elastoplastic numerical models. (to be submitted)

Results

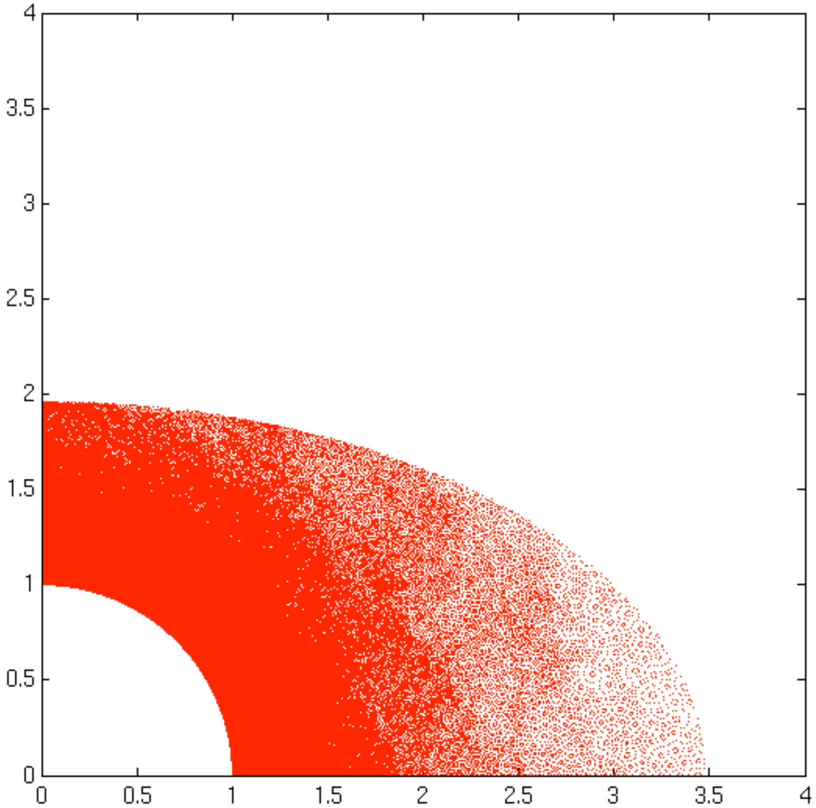

Pressure load: 1.00 Shear stress load: 0.00
4
0
0
4
0
0.5
1
Second invariant of deviatoric stress


Galin's solution
Numerical
|p|/k
P∞/k
0
0.2
0.4
0.6
0.8
1
0.5
1.5
2
2.5
3
3.5
4

Picard vs Newton-Raphson

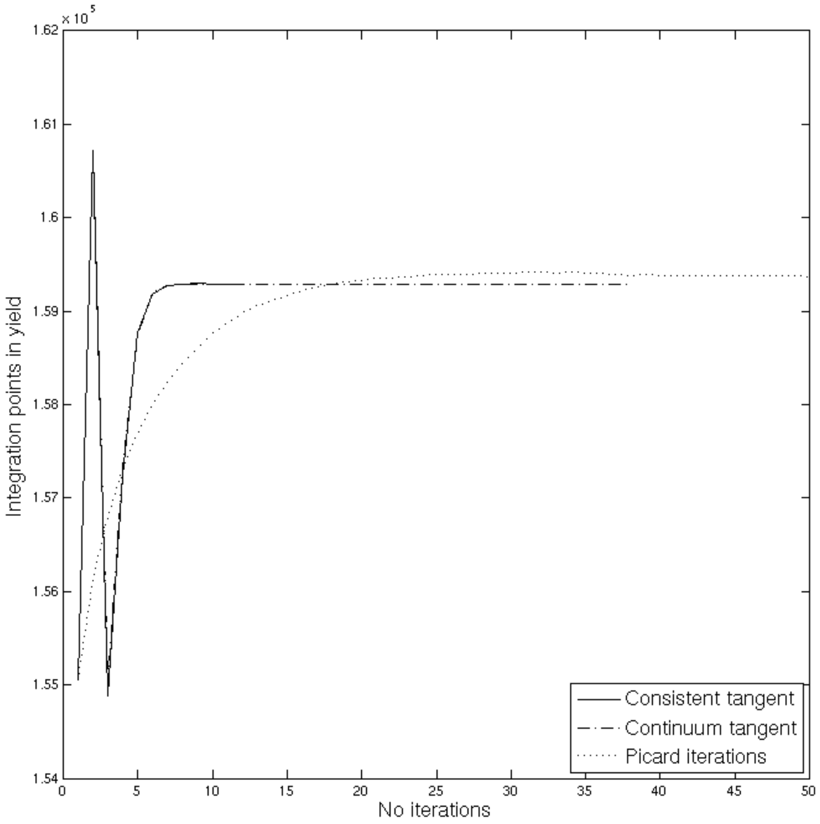


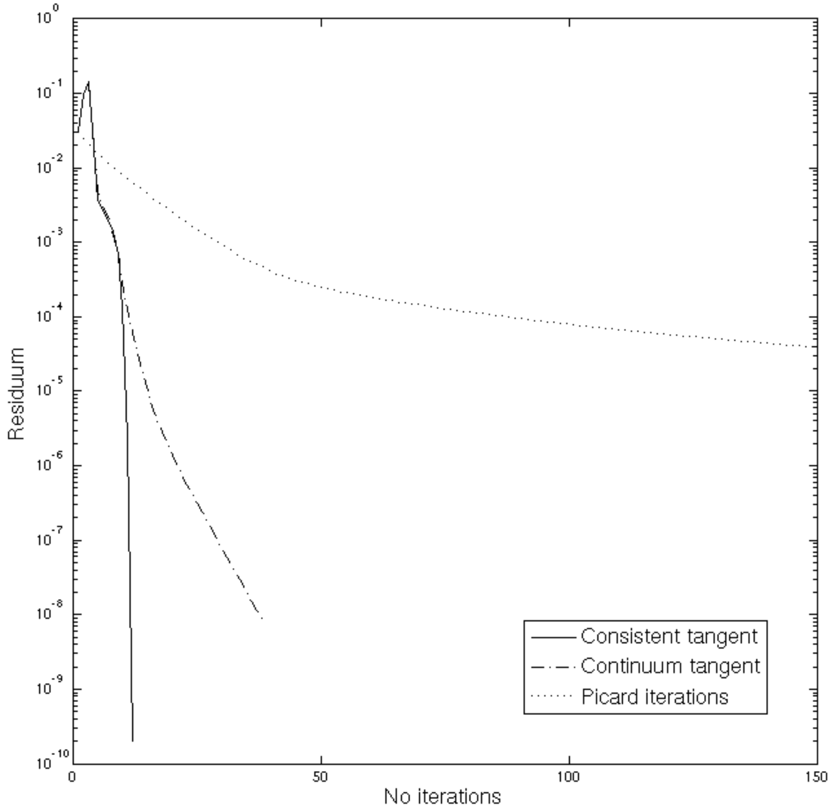
Picard: 100 iterations



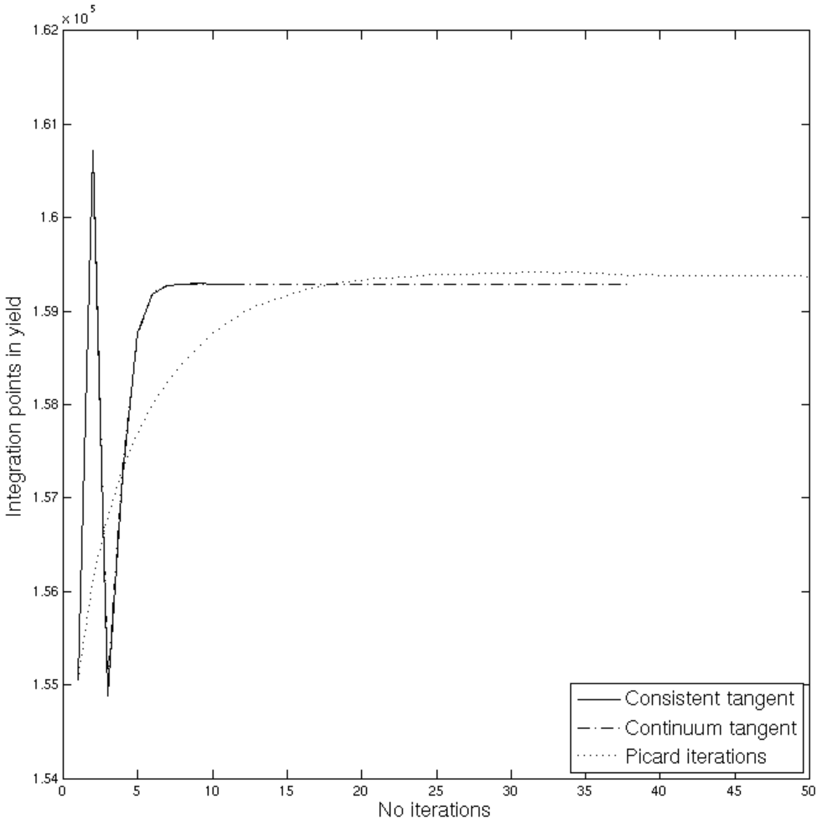
N-R: 7 iterations

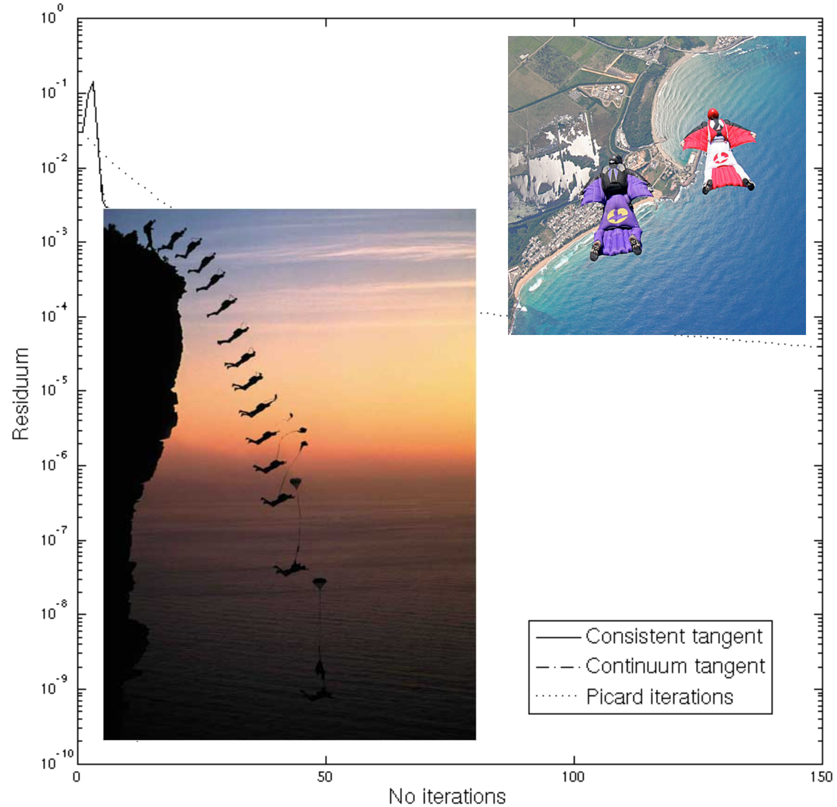
Picard vs Newton-Raphson



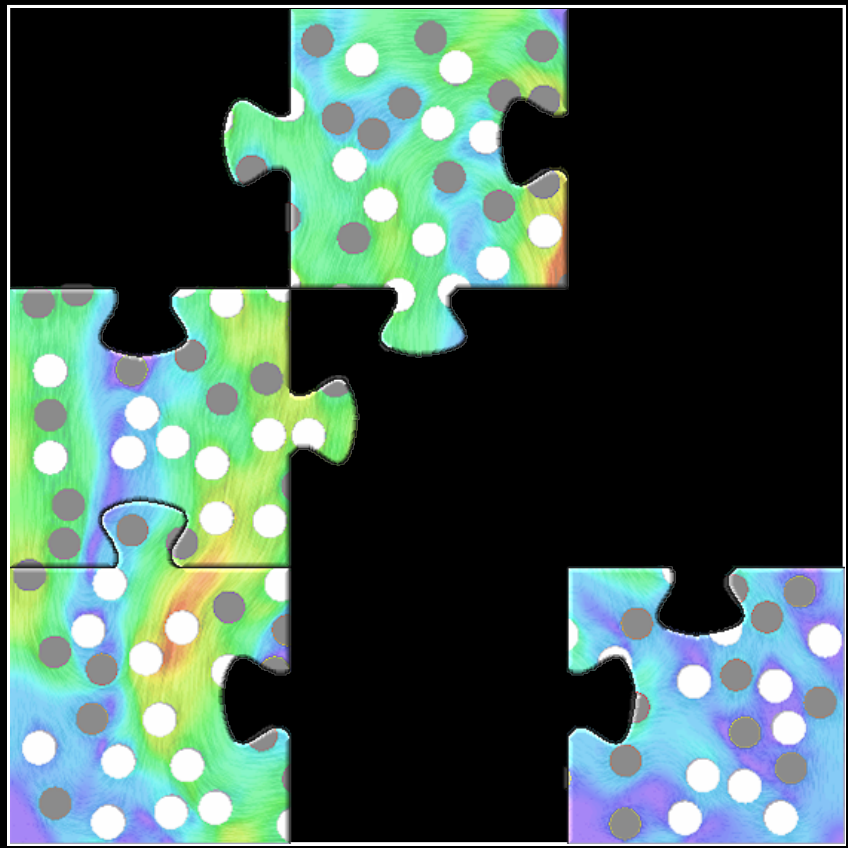


Picard vs Newton-Raphson





Newton-Raphson iterations rock



Iterative solvers

Iterative solvers

Successive approximations to the solution rather than one-shot direct approach

<u>The Conjugate Gradient method</u>

$$\min_{x_n \in K_n} \left\| x_n - x \right\|_A$$

<u>The Minimum Residual method</u>

$$\min_{x_n \in K_n} \left\| A x_n - b \right\|$$

minimization over the chain of **subspaces: n=1,..,m<ndofs**

Krylov Space

$$\mathrm{K}_n = span\{b, Ab, \ldots, A^{n-1} b\}$$

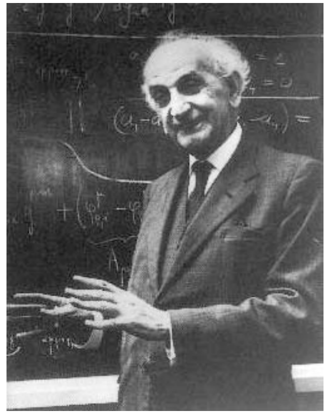
This space is intimately tied to the inverse of the matrix.

Lanczos Algorithm

$$\gamma_{i+1} v^{i+1} = A v^i - \left\langle A v^i, v^i \right\rangle v^i - \gamma_i v^{i-1}$$

$$\left\langle v^i, v^j \right\rangle = \delta_{ij}$$

Short recurrence to generate a sequence of **orthogonal vectors** in Krylov spaces for symmetric A

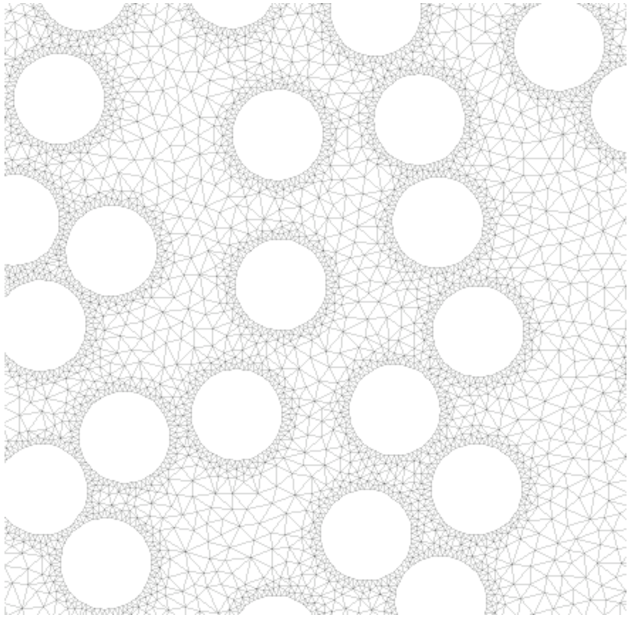


Sparse matrix – vector multiplication

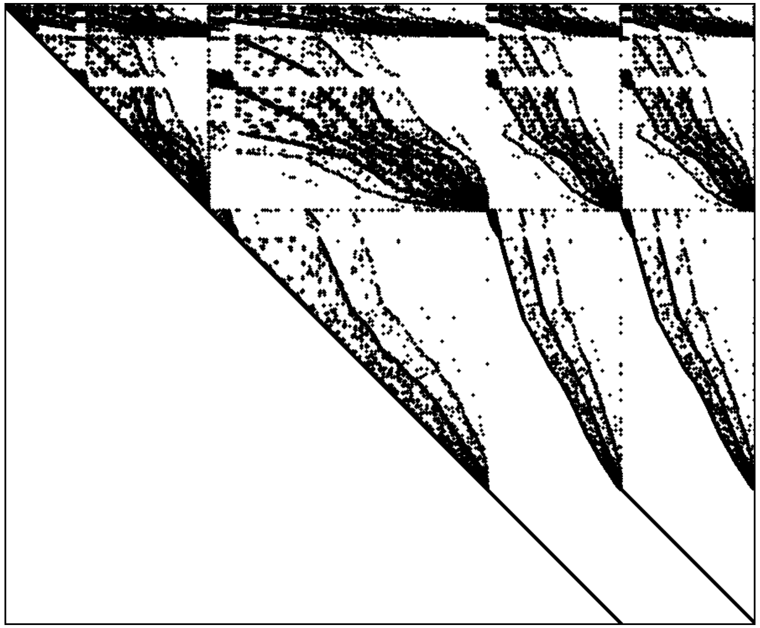
Sparse Matrix – Vector multiplication
- low FLOP to byte ratio
- *memory bounded algorithm*

$$r_i = Ax_i - b$$

Unstructured mesh



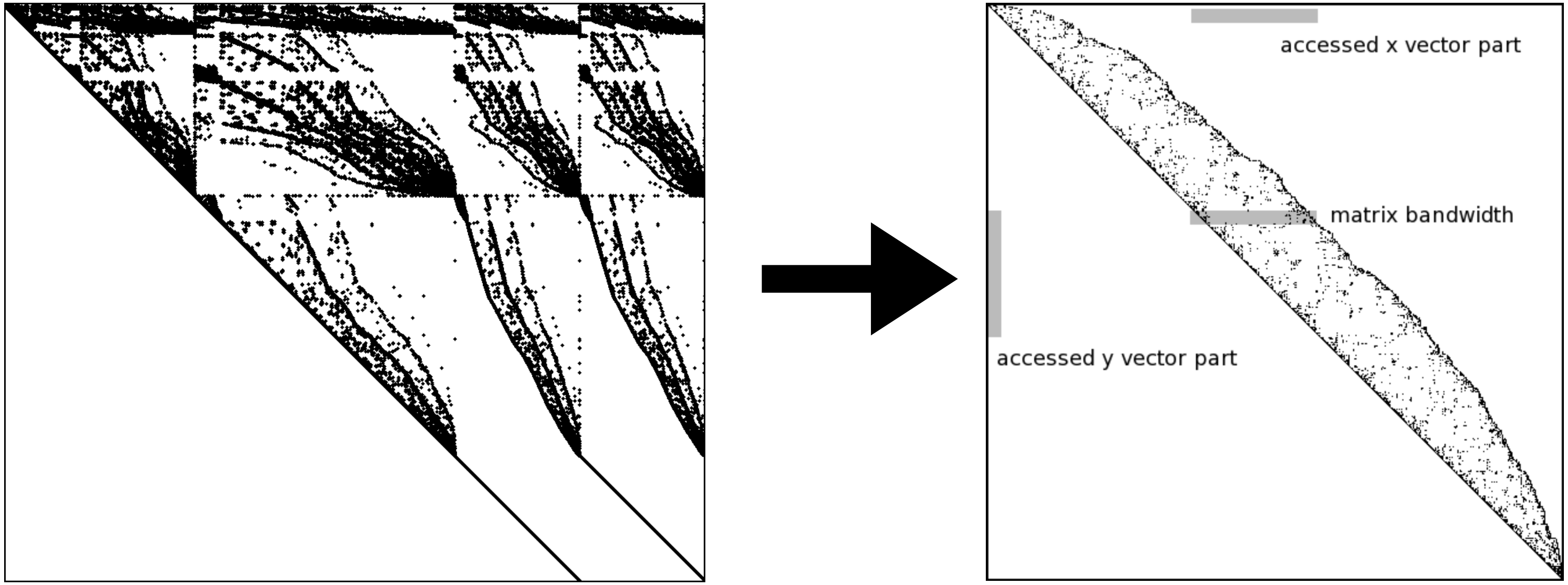
Irregular sparse pattern of A



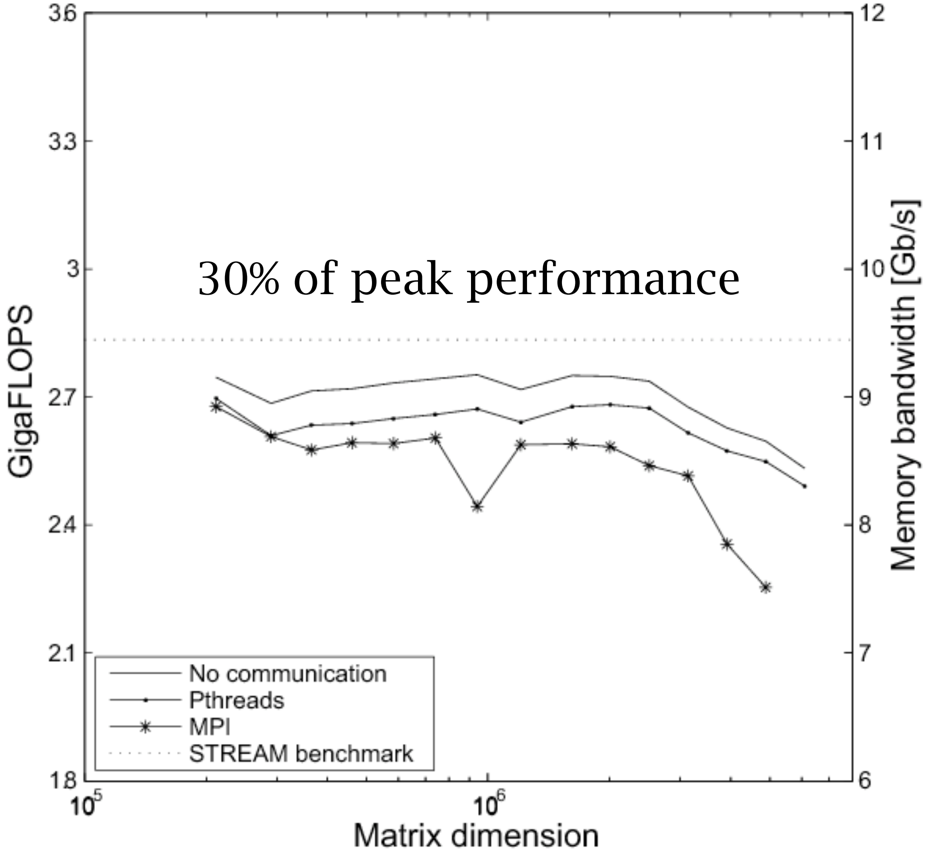
M. Krotkiewski and M. Dabrowski, **Massively parallel unstructured sparse matrix - vector product for multi-core CPUs,** *Parallel Computing, submitted*

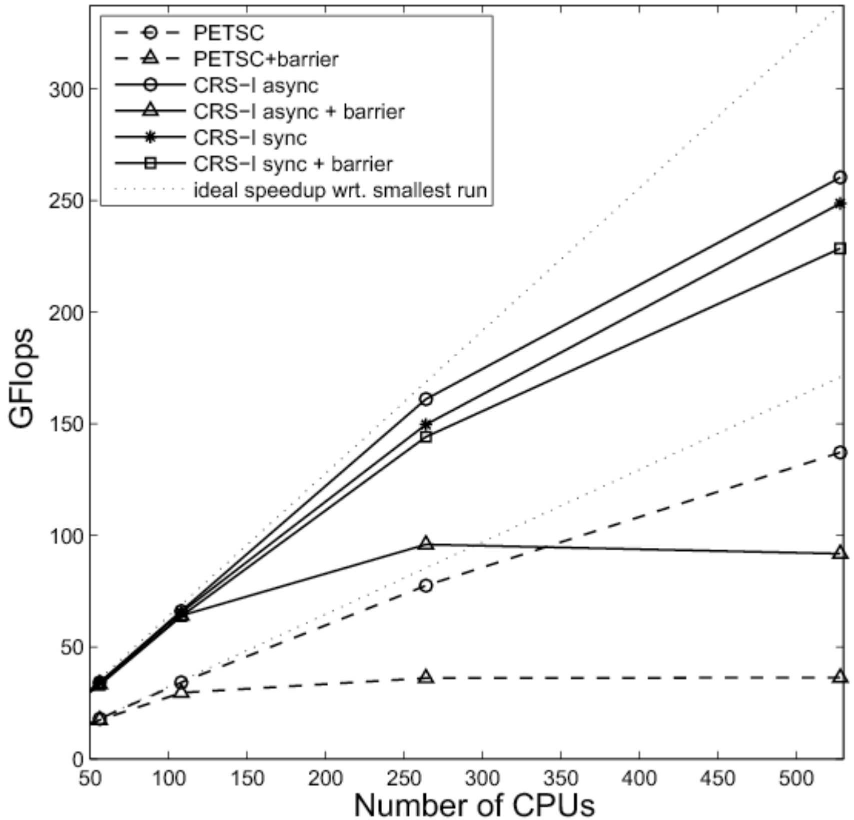
SpMV– optimizing memory bandwidth

- **cache hit ratio** (Reverse Cuthill-McKee renumbering)
- **decrease storage overhead** (blocking and symmetric storage)
- **improved memory bandwidth** (prefetching, interleaved sparse storage)



Sequential and parallel performence





ParMETIS – graph partitioning

No hybrid programming

No explicit overlapping of computation and communication

Parallel costs are mostly affected by the per-core work imbalances

ACHIEVEMENTS

- 5 TeraFLOPs (~20% of peak performance)
- Scalable up to 5400 CPUs of Cray XT4 (entire cluster)
- 800 million unknowns, 500 million elements

BRUTE FORCE ITERATIVE METHODS

Light preconditioners

100 million dofs – 1 iteration – 1000 CPUs – 0.03s (15-node tetrahedra)

MATRIX-FREE (element matrices only)

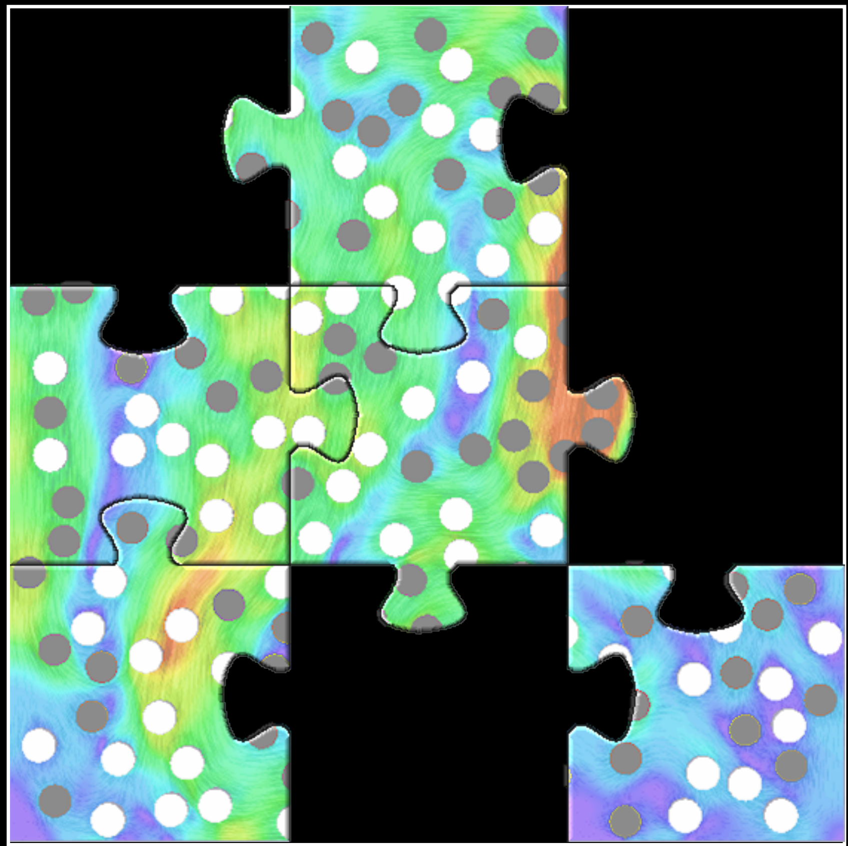
Avoiding memory bandwidth problems

More operations (up to 12x)

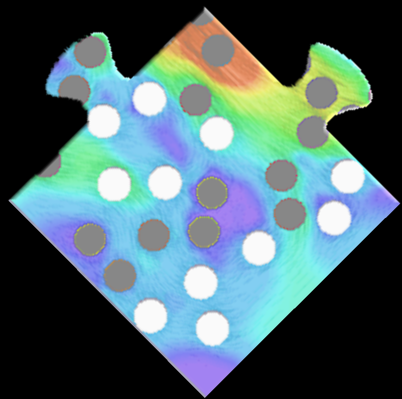
Cost of redundant computations may not be hidden

Existing implementations (~20% of peak performance)

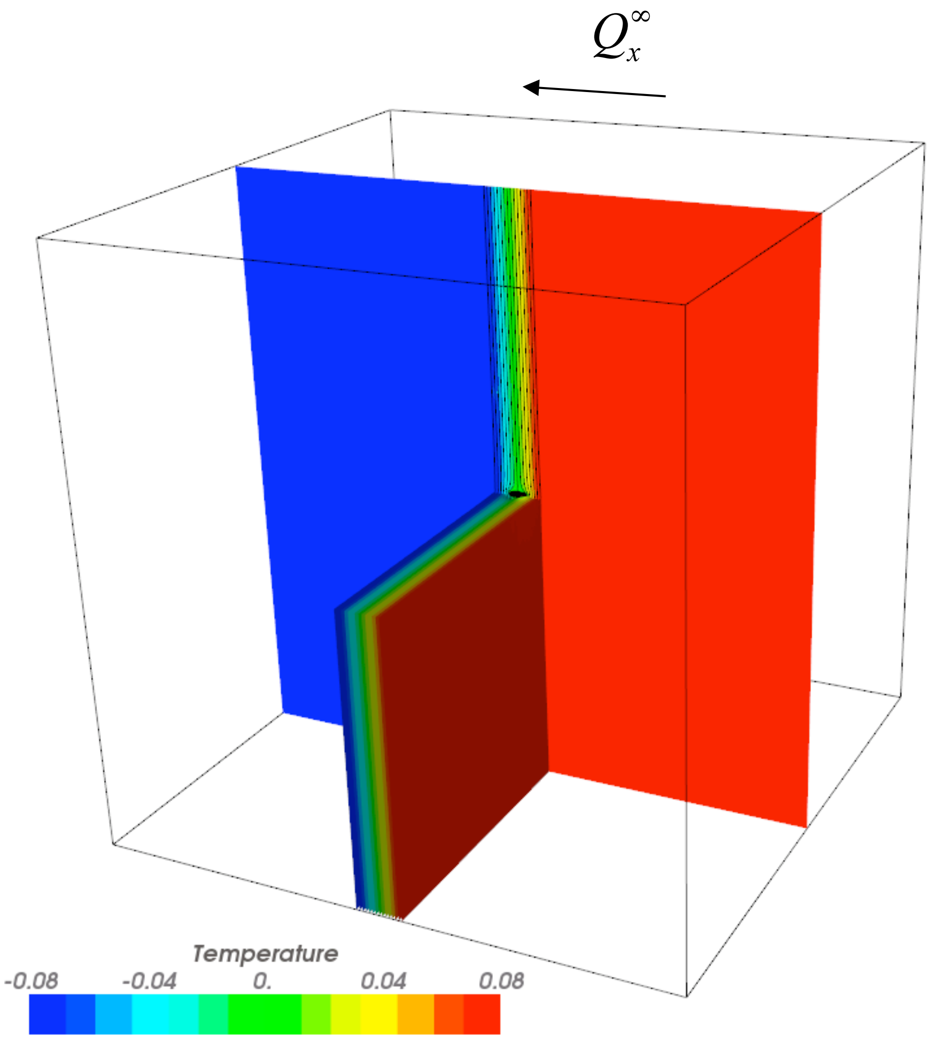
SpMV for unstructured FEM can be very efficient



Solvers – 3D benchmark



Ellipsoidal inclusion benchmark



Ellipsoidal inclusion

$$a = 3,\ b = 2,\ c = 1$$

$$k_{incl}/k_{host} \in \left[10^{-20}, 10^{20}\right]$$

$$Q_x^{incl}/Q_x^{\infty} = ?$$

Ellipsoidal inclusion - constant flux

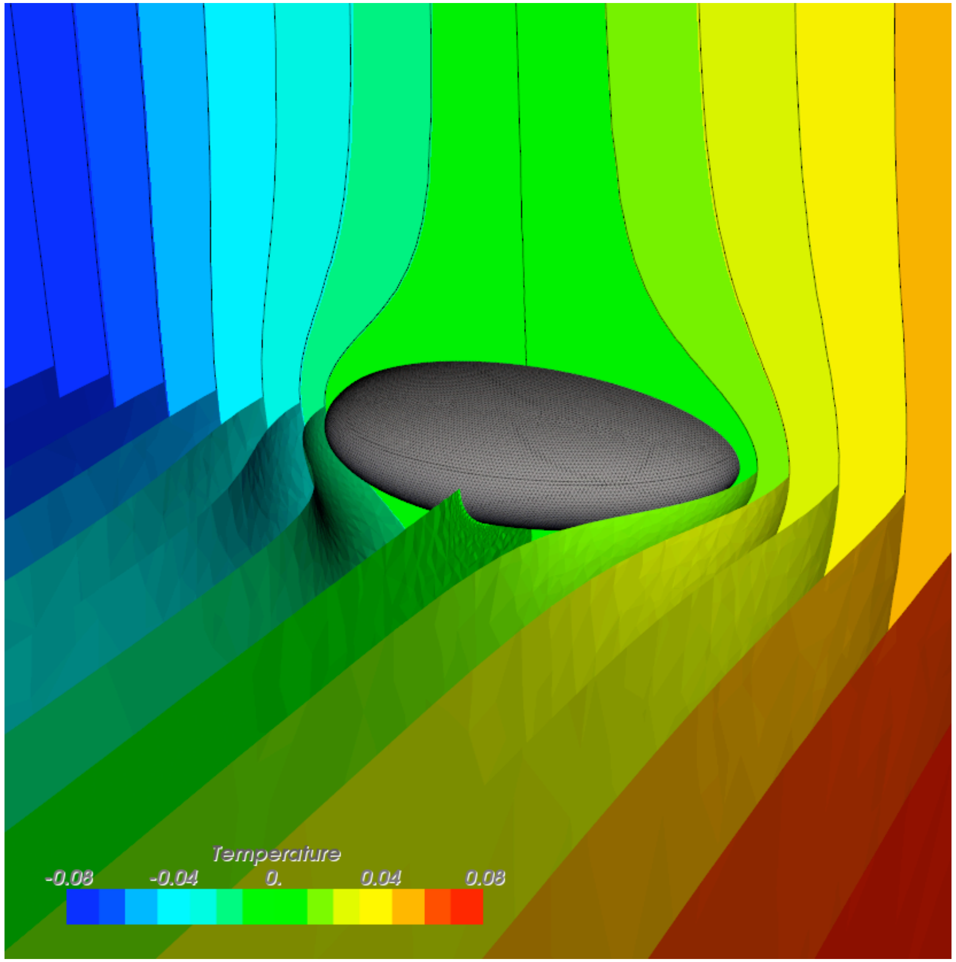
$$\frac{Q_{incl}}{Q_{\infty}} = \left(1 + \frac{k_{incl} - k_{host}}{k_{host}} I_a\right)^{-1}$$

$$I_a = \frac{abc}{\left(a^2 - b^2\right)\sqrt{a^2 - c^2}}\left(F(\theta,k) - E(\theta,k)\right)$$

$$\theta = \sin^{-1}\sqrt{1 - c^2/a^2}, \quad k = \sqrt{\frac{a^2 - b^2}{a^2 - c^2}}$$

Ellipsoidal inclusion benchmark

Q_x^{∞}



Ellipsoidal inclusion

$$a = 3,\ b = 2,\ c = 1$$

$$k_{incl}/k_{host} \in \left[10^{-20}, 10^{20}\right]$$

$$Q_x^{incl}/Q_x^{\infty} = ?$$

Ellipsoidal inclusion - constant flux

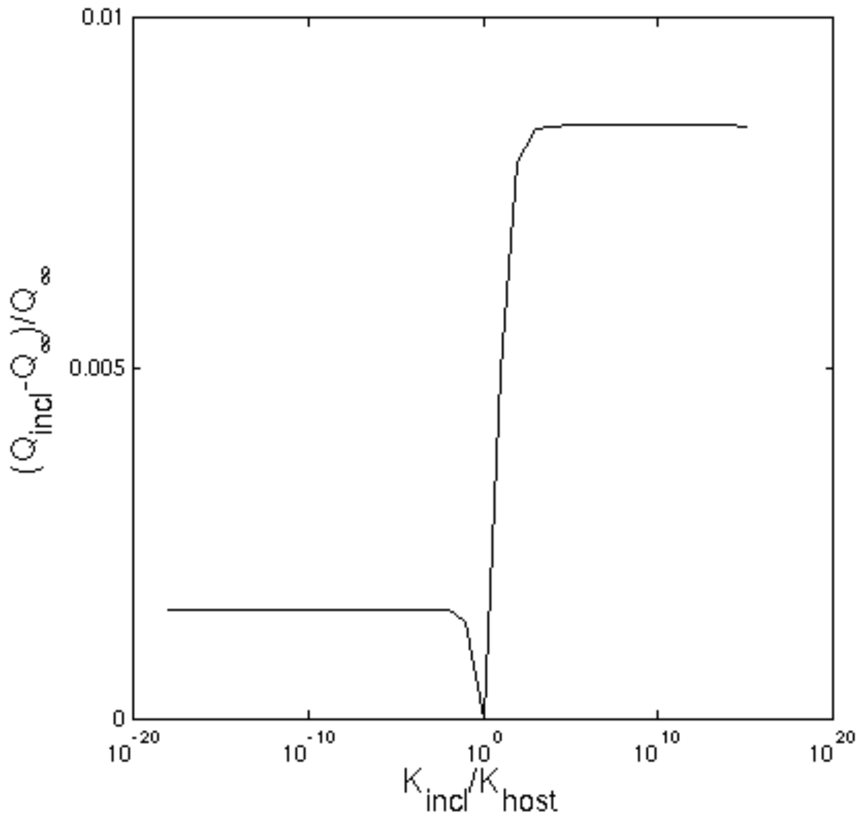
$$\frac{Q_{incl}}{Q_{\infty}} = \left(1 + \frac{k_{incl} - k_{host}}{k_{host}} I_a\right)^{-1}$$

$$I_a = \frac{abc}{\left(a^2 - b^2\right)\sqrt{a^2 - c^2}}\left(F(\theta,k) - E(\theta,k)\right)$$

$$\theta = \sin^{-1}\sqrt{1 - c^2/a^2}, \quad k = \sqrt{\frac{a^2 - b^2}{a^2 - c^2}}$$

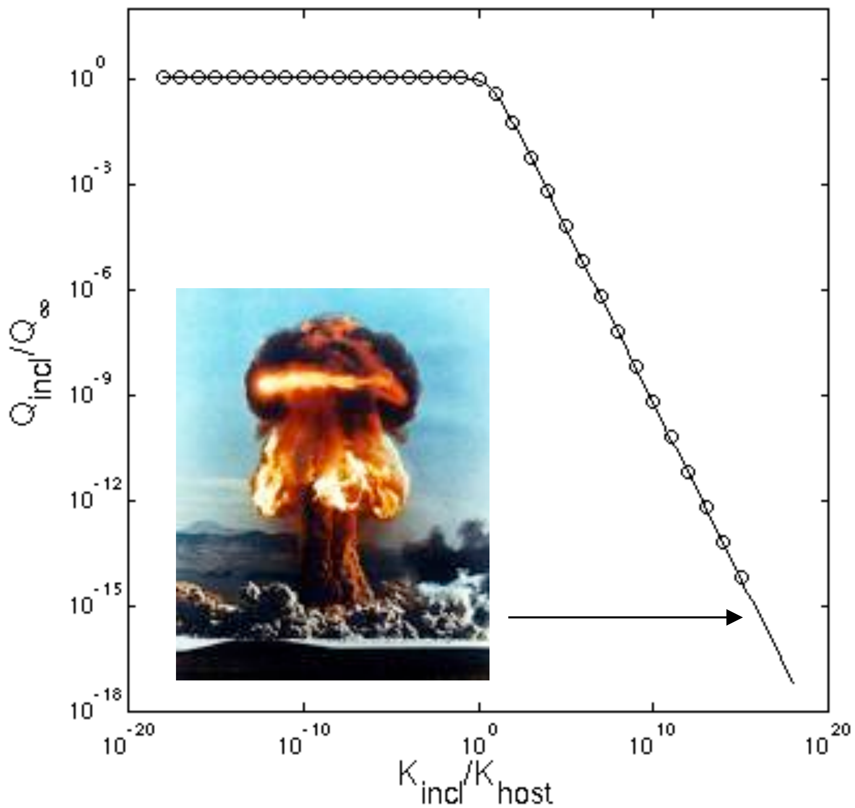
Analytical results vs direct solver

Flux error

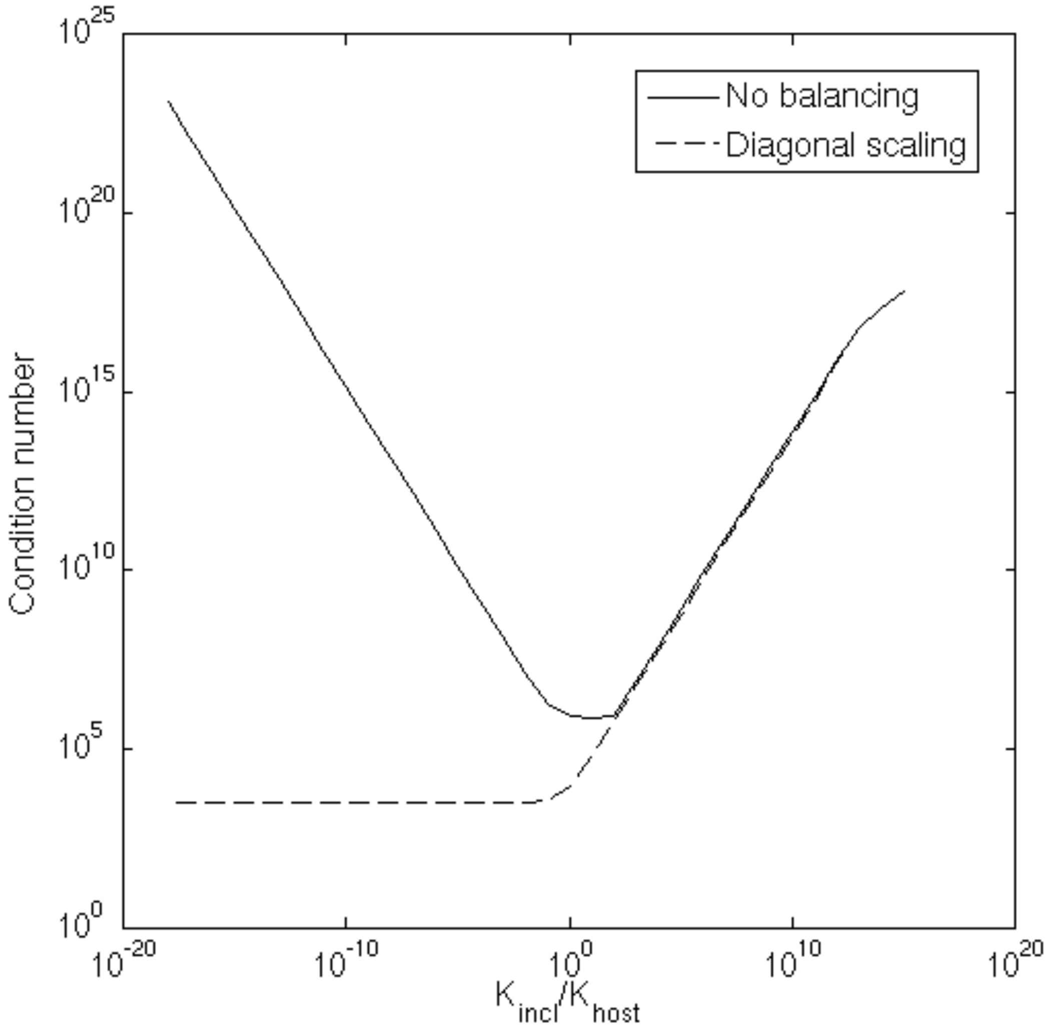


0.5 Mnodes, a/h = 0.03

Flux



Cholesky: 10^12 operations
1 CPU – 10 min



Condition number

$$K(A) = \|A^{-1}\|\|A\| = \lambda_{\max}/\lambda_{\min}$$

Relative error

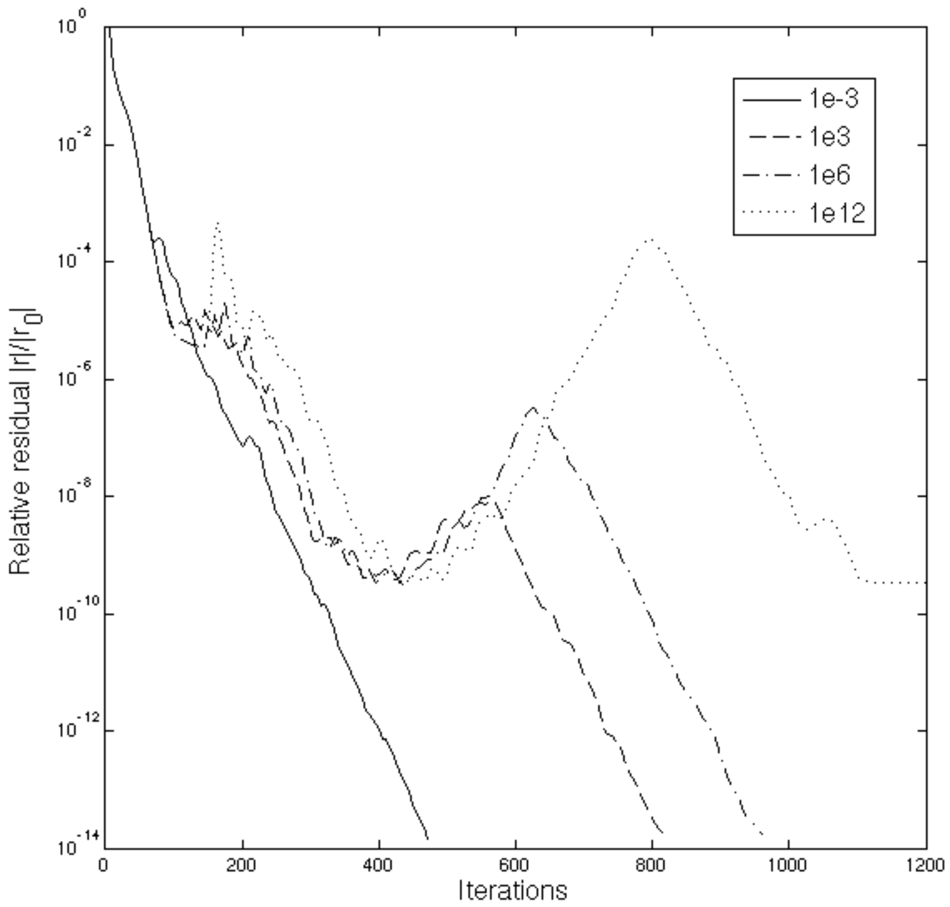
$$\|\tilde{x} - x\|/\|x\| \leq \rho \varepsilon K(A)$$

Cholesky factorization

$$\rho \approx 100$$

$$\varepsilon = 10^{-18}$$

PCG – relative residual



<u>Storage</u>

~ndofs

<u>Operation count</u>

nnzA ~ 8 x ndofs

4 million x 2 operations x 1000 it

8 G operations

10 sec

Cholesky: 1 T operations (10 min)

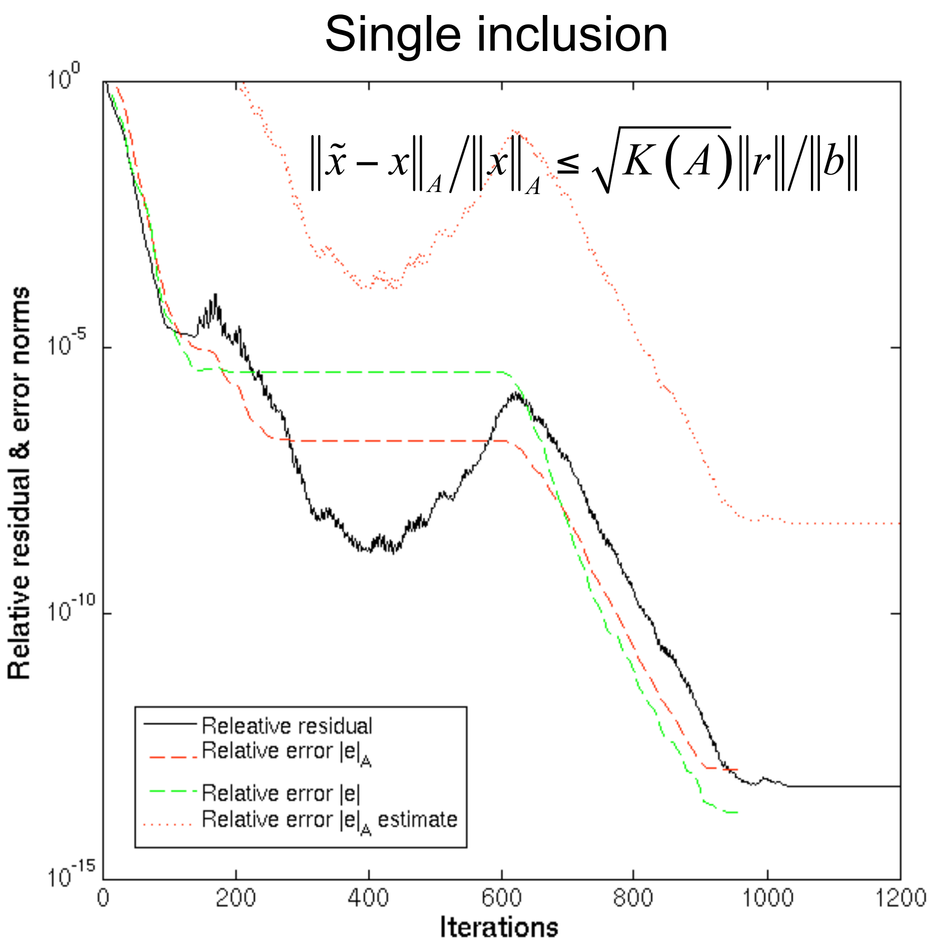
<u>Condition number</u>

Increase in number of iterations

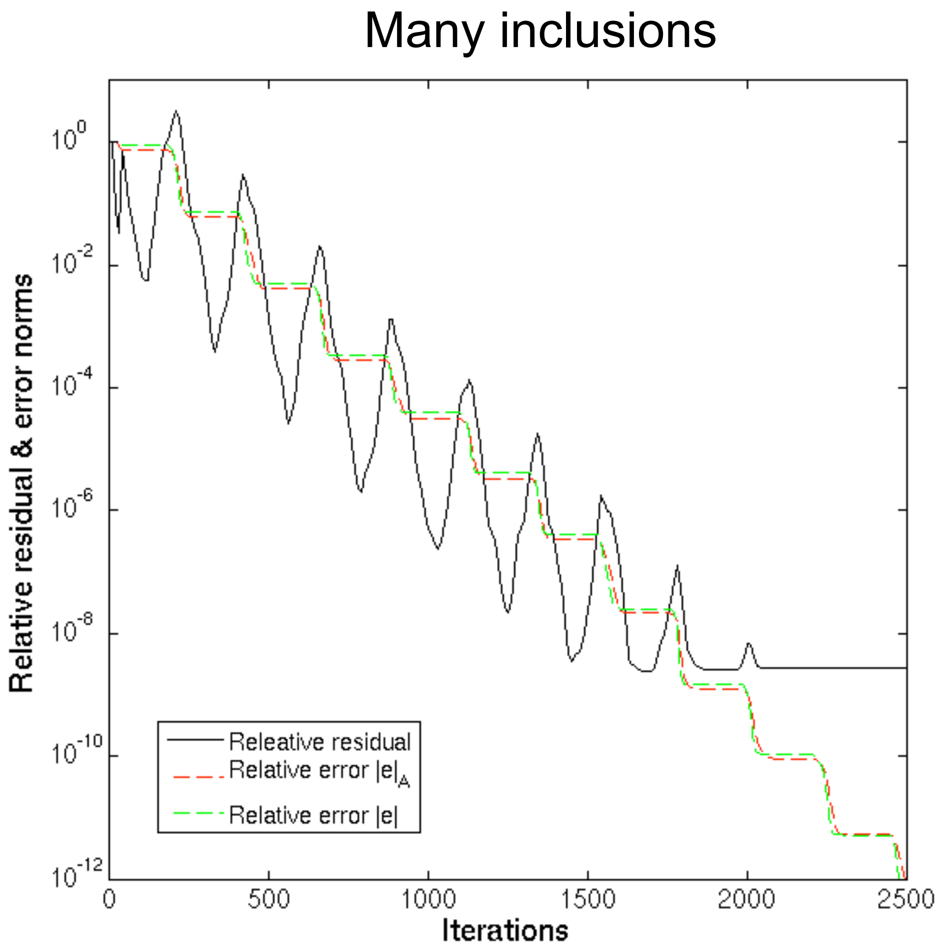
Stopping criteria

Rounding-off errors

When to stop?

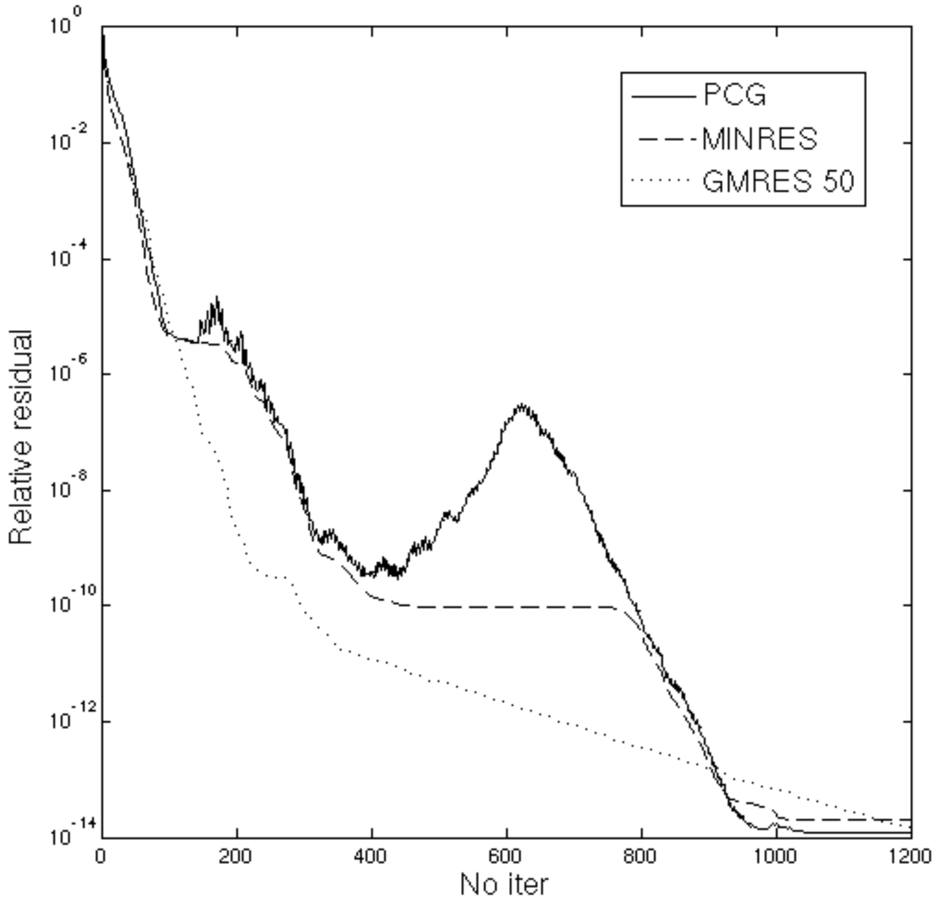


Single inclusion
$\|\tilde{x}-x\|_A / \|x\|_A \le \sqrt{K(A)}\,\|r\|/\|b\|$
Relative residual & error norms
Iterations
Releative residual
Relative error |e|A
Relative error |e|
Relative error |e|A estimate



Many inclusions
Relative residual & error norms
Iterations
Releative residual
Relative error |e|A
Relative error |e|

Rounding-off errors



Lanczos method

$$\gamma_{i+1} v^{i+1} = A v^i - \left\langle A v^i , v^i \right\rangle v^i - \gamma_i v^{i-1}$$

$$\left\langle v^i , v^j \right\rangle = \delta_{ij}$$

Seven Deadly Sins of Numerical Computing

by BJ McCartin

- Algorithmic Idolatry
- Using an Inappropriate Model
- **Temptation From Ill-Conditioning**
- **Ruination by Rounding**
- Numerical Philistinism
- Central Differencing Singular Perturbations
- Uncritical Use of Non-orthogonal Mappings

PRECONDITIONERS

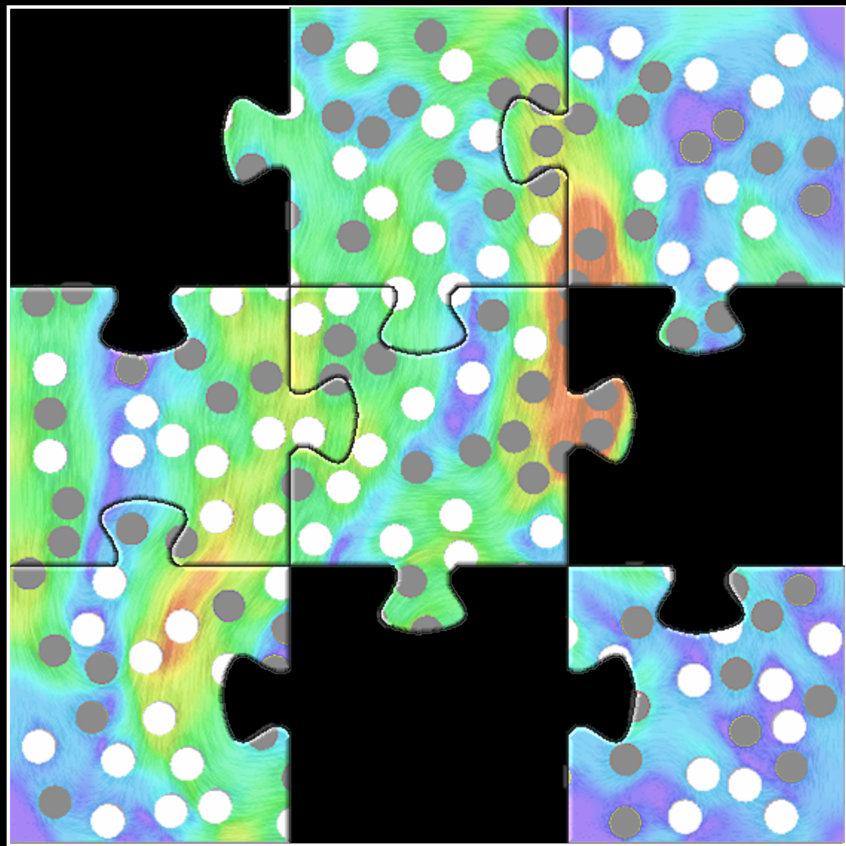
improve spectral properties

$$P^{-1} A x_i = P^{-1} b$$

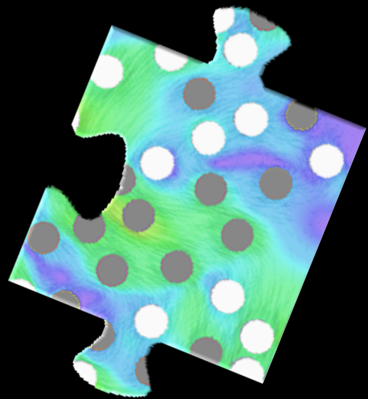
increased convergence rate
less iterations - less rounding

stopping criteria, error

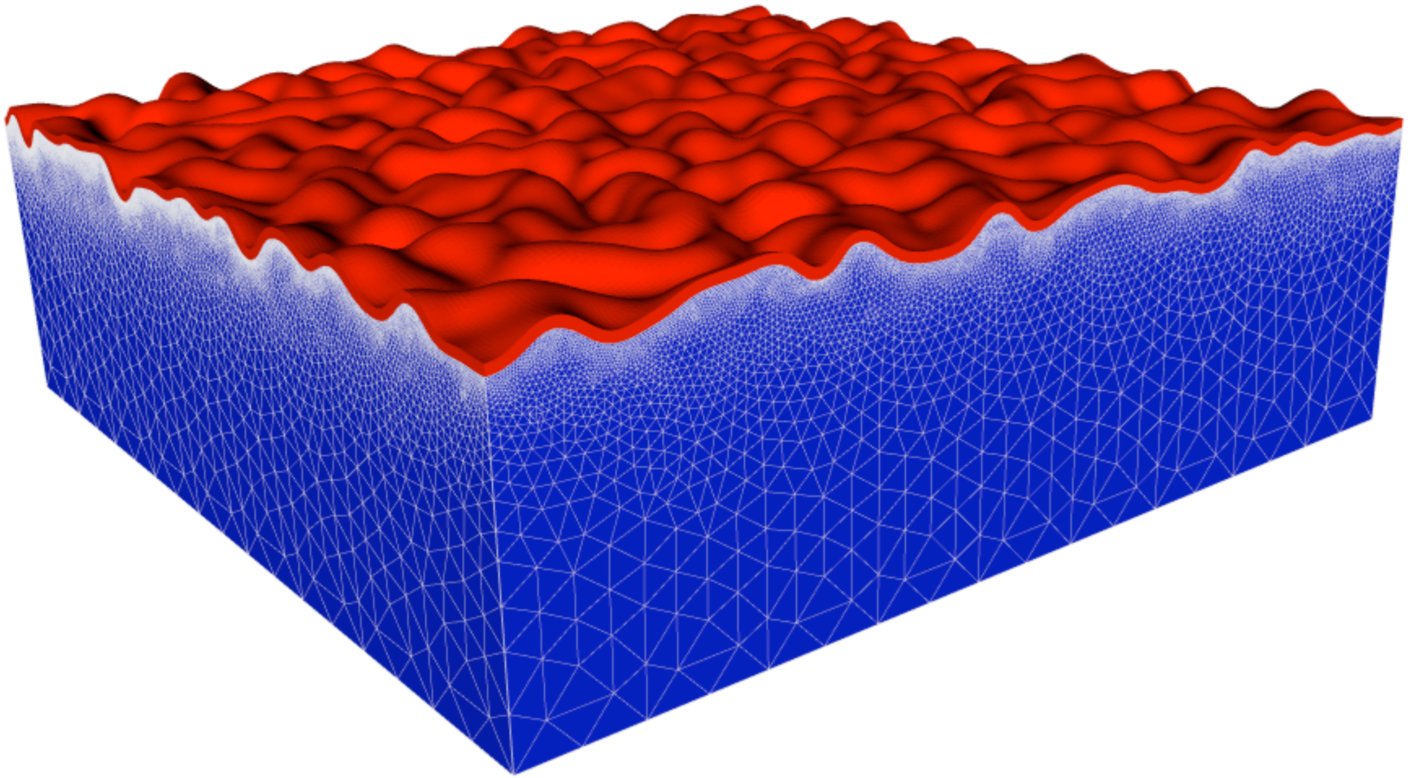
Iterative solvers are necessary in 3D
Don't get tempted from ill-conditioning!



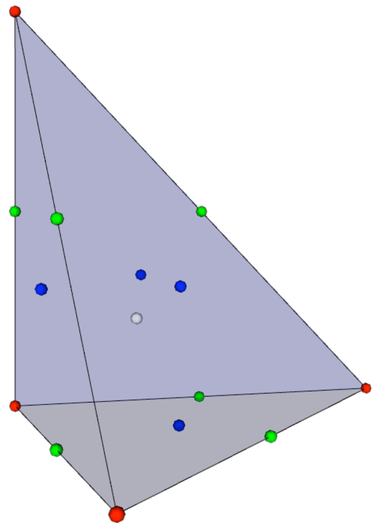
Applications in 3D



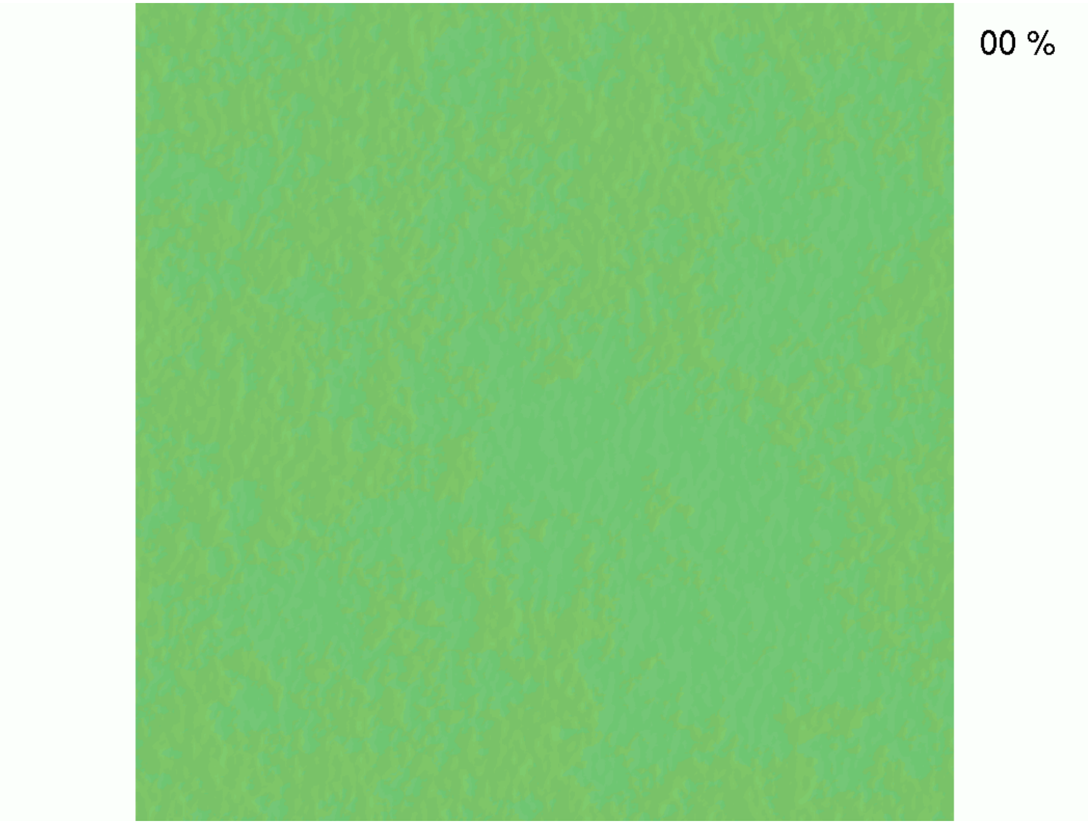
3D Folding



30 million dofs
2 million elements
512 CPUs
500s / time step
MINRES solver



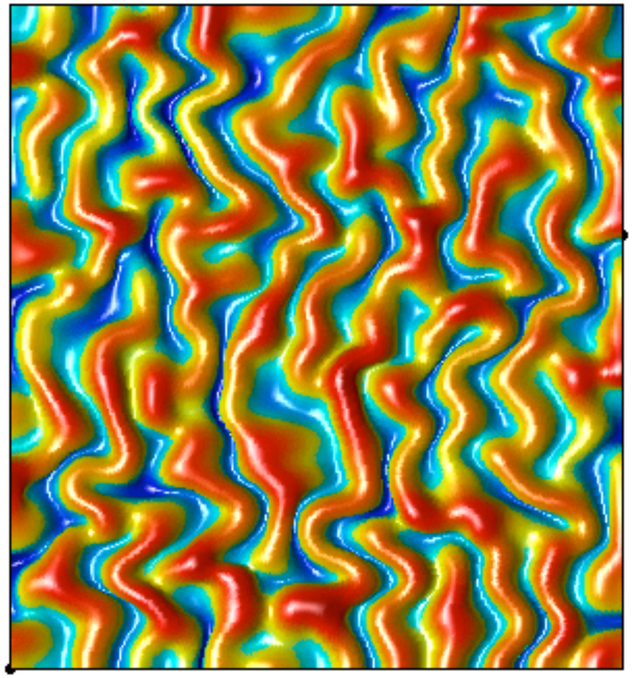
Evolution of large amplitude 3D fold patterns: a FEM study, Schmid, D. W.; Dabrowski, M.; Krotkiewski, M.; Physics of the Earth and Planetary Interiors, Volume 171, Issue 1-4, p. 400-408.



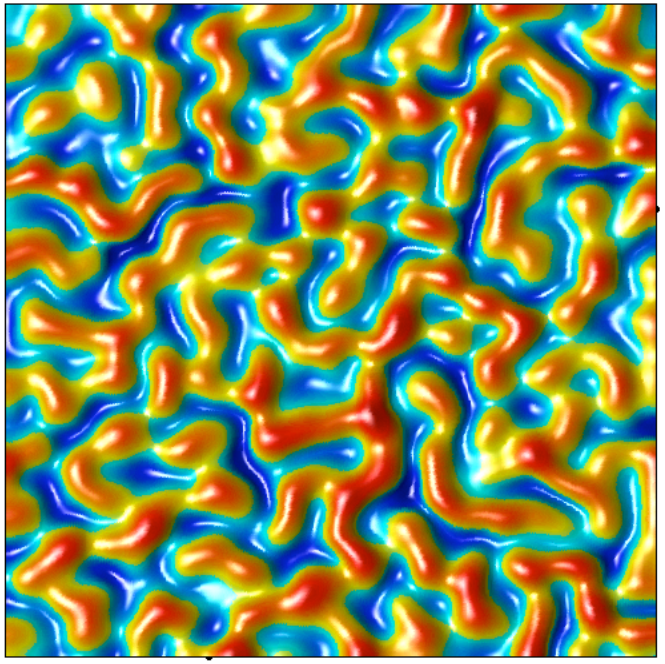
Top-view
Color-coded elevation
Viscosity ratio 200
Constriction (up to 40%)

Folding – pure shear vs superposition

Superposed



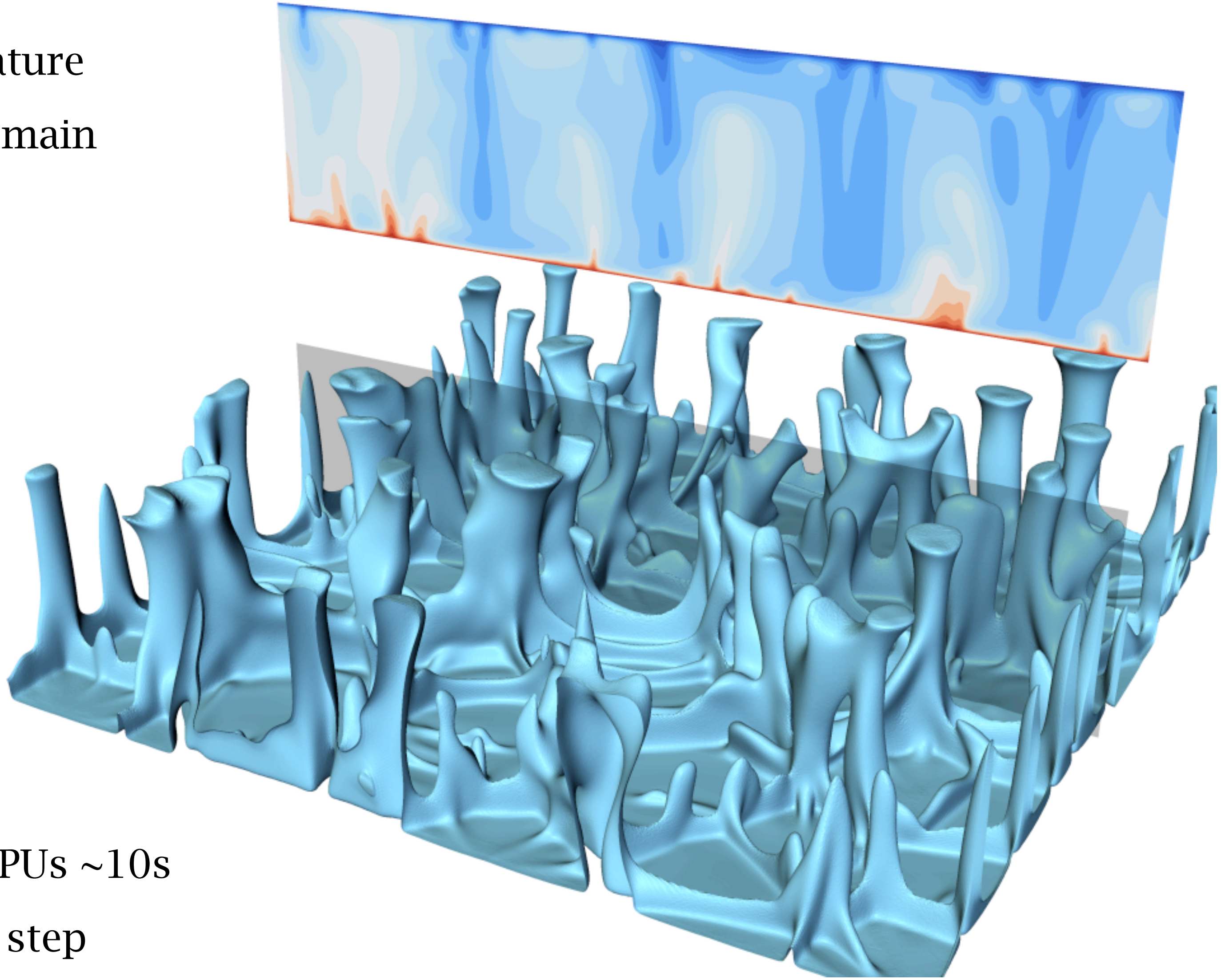
Constriction



Porous convection

Iso-surface of temperature

2D cut through the domain



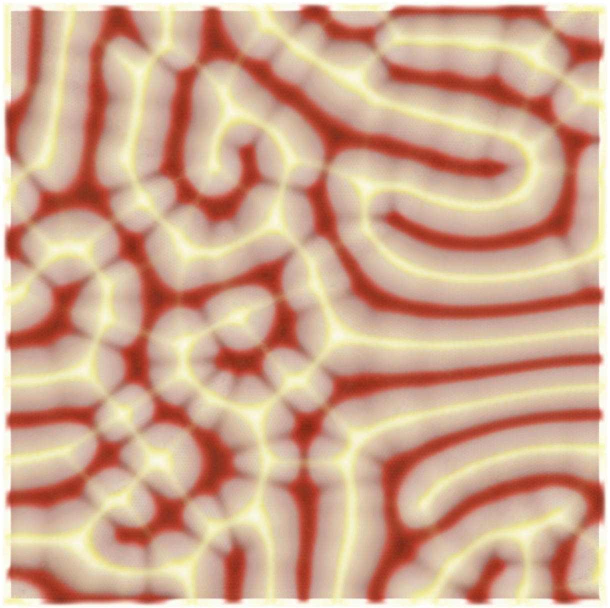
150 million elements

1 time step on 1024 CPUs ~10s

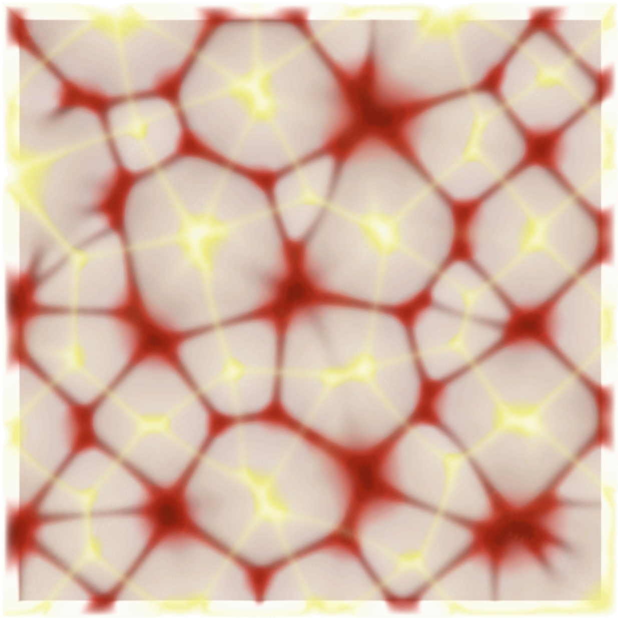
Data size ~5GB / time step

3D Porous convection - patterns

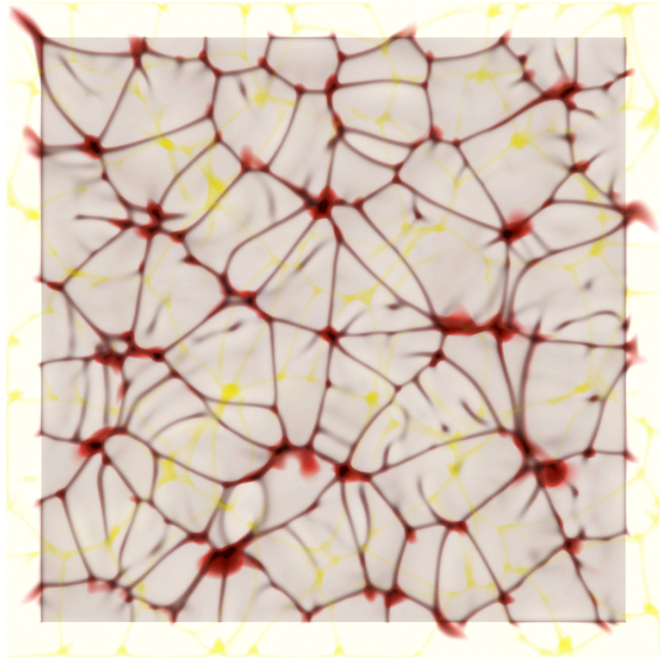
Ra=100



Ra=200



Ra=1000



Thank you!