HPDDM une bibliothèque haute performance unifiée pour les méthodes de décomposition de domaine

P. Jolivet and Frédéric Nataf

Laboratory J.L. Lions, Univ. Paris VI, Equipe Alpines INRIA-LJLL et CNRS

joint work with **R. Haferssas, F. Hecht, P.H. Tournier,** (Paris VI)

TERATEC 2015

Motivation: Scientific computing

Large discretized system of PDEs
strongly heterogeneous coefficients
(high contrast, multiscale)

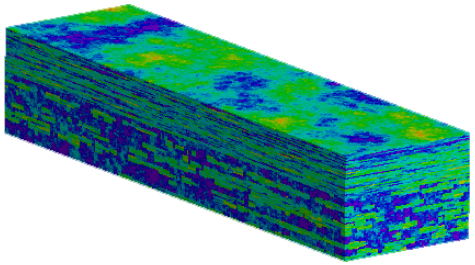
E.g. Darcy pressure equation,
P^1-finite elements:

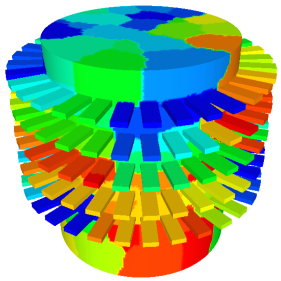
$$\mathbf{A}\mathbf{u} = \mathbf{f}$$

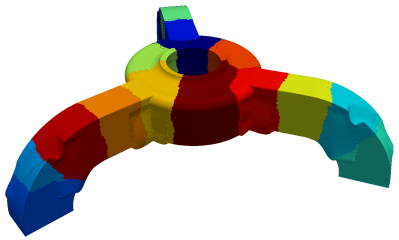
$$\text{cond}(\mathbf{A}) \sim \frac{\kappa_{\max}}{\kappa_{\min}} h^{-2}$$

Goal:
iterative solvers
robust in size and heterogeneities

Applications:
flow in heterogeneous /
stochastic / layered media
structural mechanics
electromagnetics
etc.







80% of the elapsed time for typical engineering applications

Changes in hardware

Since year 2005:

- CPU frequency stalls at 2-3 GHz due to the heat dissipation wall.
  The only way to improve the performance of computer is to go parallel

  Intel calls the speed/power tradeoff a "fundamental theorem of multicore processors"

- Power consumption is an issue:
  - Large machines (hundreds of thousands of cores) cost 10-15% of their price in energy every year.
  - Smartphone, tablets, laptops (quad - octo cores) have limited power supplies

All fields of computer science are impacted.

Where to make effort in scientific computing

to compute right

- in the past: Numerical analysis of discretization schemes, a posteriori error estimates, mesh generation, reduced basis method
- Now: business as usual

to compute faster

- in the past: invest in a new machine every three years
- Now: invest every five years and add an investment in algorithmic research:

to use less energy

- in the past: nobody cared
- Now: communication avoiding algorithms

Need for parallel linear solvers

A simplified view of modern architectures

- Unlimited number of fast cores
- Distributed data
- Limited amount of slow and energy intensive communication

Coarse Grain algorithm

- Maximize local computations
- Minimize communications (saves time and energy altogether)
- No sequential task

$Au = f$? Panorama of linear solvers

Direct Solvers

MUMPS (J.Y. L'Excellent), SuperLU (Demmel, . . .), PastiX, UMFPACK, PARDISO (O. Schenk),

Iterative Methods

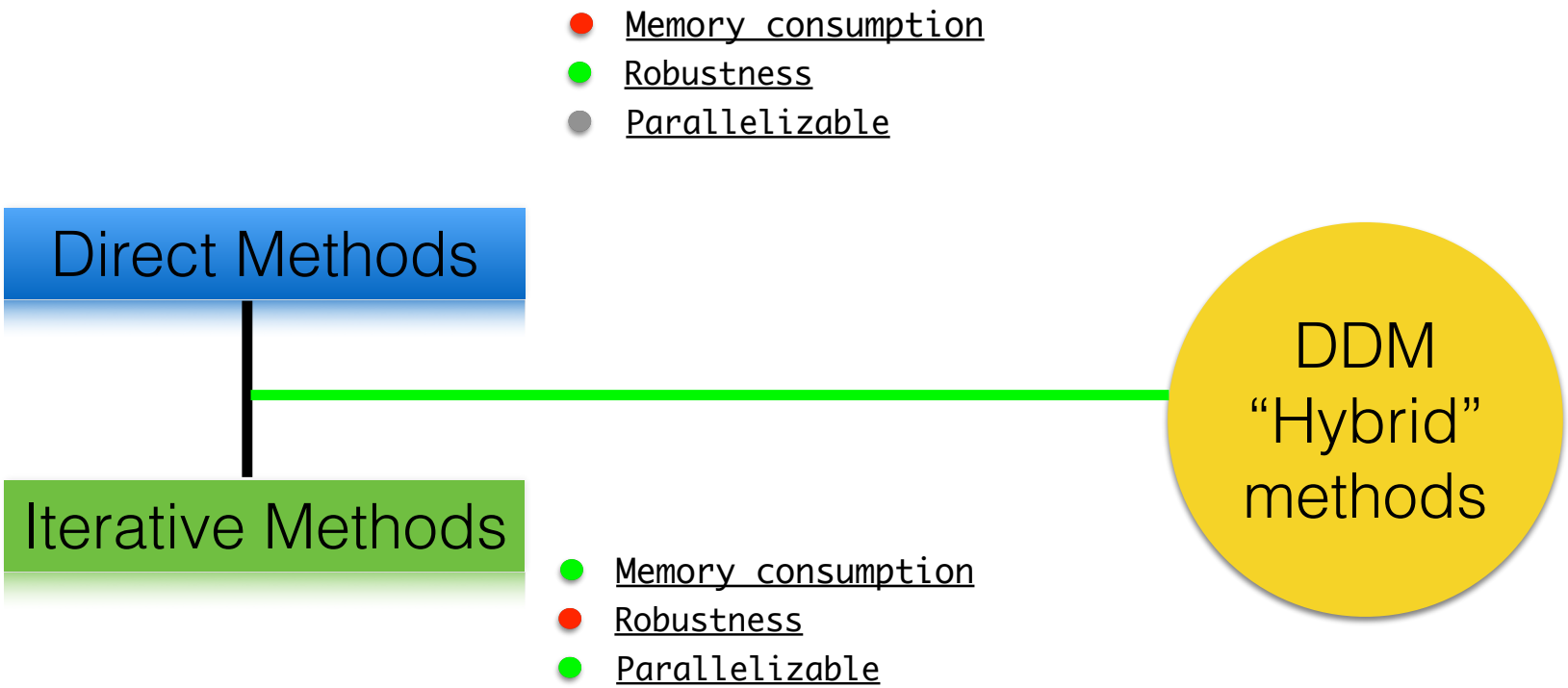
- Fixed point iteration: Jacobi, Gauss-Seidel, SSOR
- Krylov type methods: Conjuguate Gradient (Stiefel-Hestenes), GMRES (Y. Saad), QMR (R. Freund), MinRes, BiCGSTAB (van der Vorst)

"Hybrid Methods"

- Multigrid (A. Brandt, Ruge-Stüben, Falgout, McCormick, A. Ruhe, Y. Notay, . . .)
- Domain decomposition methods (O. Widlund, C. Farhat, J. Mandel, P.L. Lions,) are a naturally parallel compromise

Why Domain Decomposition Methods ?

How can we solve a large sparse system $\boldsymbol{A}u = F \in \mathbb{R}^n$?



A short introduction to DDM

Consider the linear system: $Au = f \in \mathbb{R}^n$.

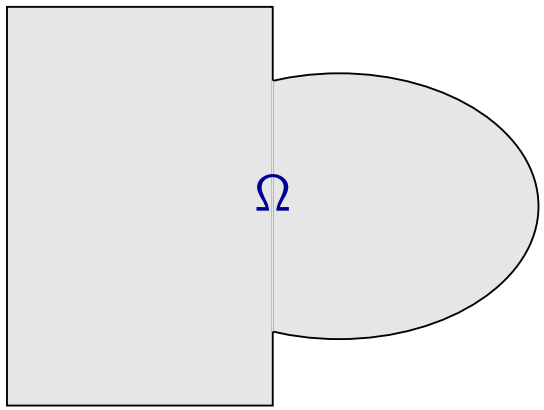
Given a decomposition of $[\![1; n]\!]$, $(\mathcal{N}_1, \mathcal{N}_2)$, define:

- the restriction operator R_i from $\mathbb{R}^{[\![1;n]\!]}$ into $\mathbb{R}^{\mathcal{N}_i}$,
- R_i^T as the extension by 0 from $\mathbb{R}^{\mathcal{N}_i}$ into $\mathbb{R}^{[\![1;n]\!]}$.

Then solve concurrently:

$$u_1^{m+1} = u_1^m + A_1^{-1} R_1 (f - Au^m) \qquad u_2^{m+1} = u_2^m + A_2^{-1} R_2 (f - Au^m)$$

where $u_i^m = R_i u^m$ and $A_i := R_i A R_i^T$.



A short introduction to DDM

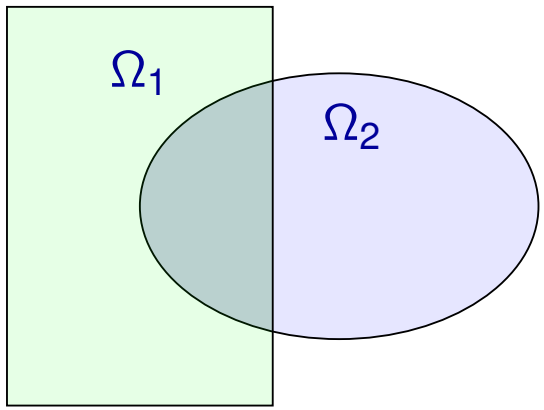
Consider the linear system: $Au = f \in \mathbb{R}^n$.
Given a decomposition of $[\![1; n]\!]$, $(\mathcal{N}_1, \mathcal{N}_2)$, define:

- the restriction operator R_i from $\mathbb{R}^{[\![1;n]\!]}$ into $\mathbb{R}^{\mathcal{N}_i}$,
- R_i^T as the extension by 0 from $\mathbb{R}^{\mathcal{N}_i}$ into $\mathbb{R}^{[\![1;n]\!]}$.

Then solve concurrently:

$$u_1^{m+1} = u_1^m + A_1^{-1} R_1 (f - Au^m) \qquad u_2^{m+1} = u_2^m + A_2^{-1} R_2 (f - Au^m)$$

where $u_i^m = R_i u^m$ and $A_i := R_i A R_i^T$.



A short introduction to DDM

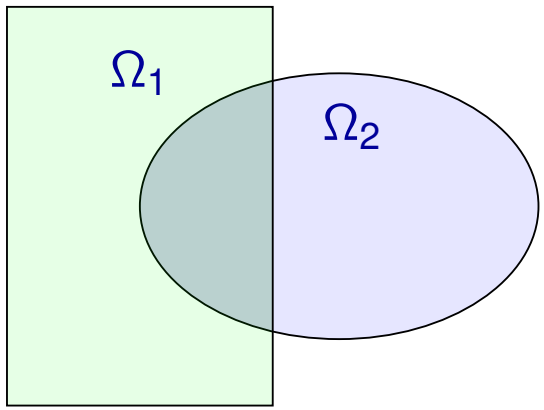
Consider the linear system: $Au = f \in \mathbb{R}^n$.
Given a decomposition of $[\![1; n]\!]$, $(\mathcal{N}_1, \mathcal{N}_2)$, define:

- the restriction operator R_i from $\mathbb{R}^{[\![1;n]\!]}$ into $\mathbb{R}^{\mathcal{N}_i}$,
- R_i^T as the extension by 0 from $\mathbb{R}^{\mathcal{N}_i}$ into $\mathbb{R}^{[\![1;n]\!]}$.

Then solve concurrently:

$$u_1^{m+1} = u_1^m + A_1^{-1} R_1 (f - Au^m) \qquad u_2^{m+1} = u_2^m + A_2^{-1} R_2 (f - Au^m)$$

where $u_i^m = R_i u^m$ and $A_i := R_i A R_i^T$.

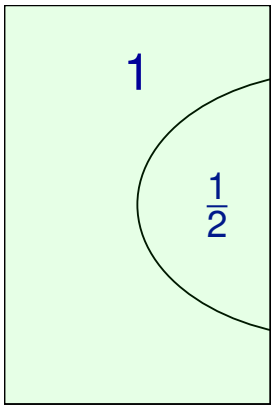


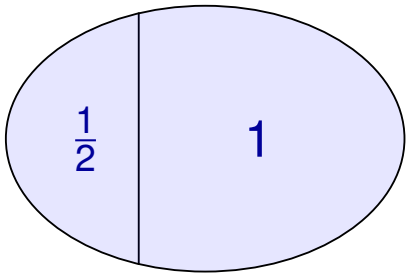
A short introduction II

We have effectively divided, but we have yet to conquer.

Duplicated unknowns coupled via a *partition of unity*:

$$I = \sum_{i=1}^{N} R_i^T D_i R_i.$$





Then, $u^{m+1} = \sum_{i=1}^{N} R_i^T D_i u_i^{m+1}$.

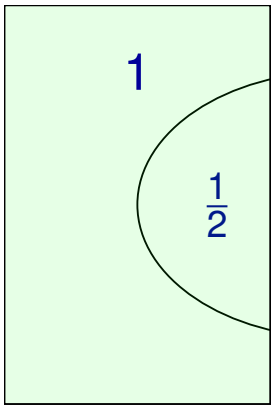
$$M_{RAS}^{-1} = \sum_{i=1}^{N} R_i^T D_i A_i^{-1} R_i.$$

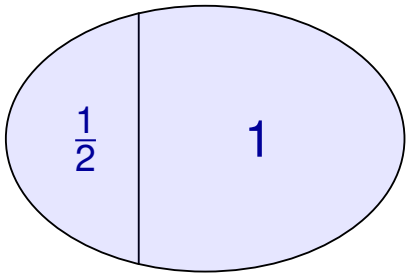
A short introduction II

We have effectively divided, but we have yet to conquer.

Duplicated unknowns coupled via a *partition of unity*:

$$I = \sum_{i=1}^{N} R_i^T D_i R_i.$$





Then, $u^{m+1} = \sum_{i=1}^{N} R_i^T D_i u_i^{m+1}$.

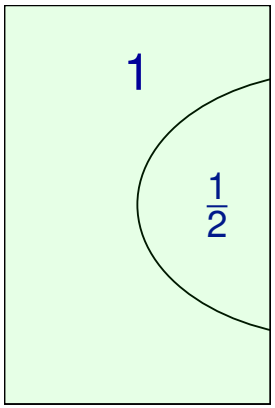
$$M_{RAS}^{-1} = \sum_{i=1}^{N} R_i^T D_i A_i^{-1} R_i.$$

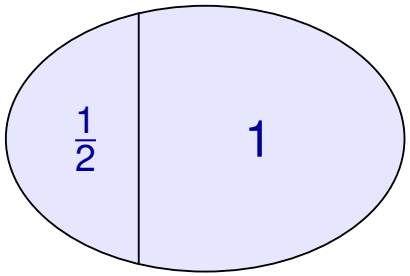
A short introduction II

We have effectively divided, but we have yet to conquer.

Duplicated unknowns coupled via a *partition of unity*:

$$I = \sum_{i=1}^{N} R_i^T D_i R_i.$$





Then, $u^{m+1} = \sum_{i=1}^{N} R_i^T D_i u_i^{m+1}$.

$$M_{RAS}^{-1} = \sum_{i=1}^{N} R_i^T D_i A_i^{-1} R_i.$$

Why One-Level methods are not so good?

		AS	RAS	ORAS	SORAS
Nbr DOFs	Nbr subdom	iteration	iteration	iteration	iteration
66703	16	140	140	96	57
130433	32	245	216	146	109
266812	64	383	312	245	203
541838	128	566	460	480	398

Table: 2D Elasticity: number of of GMRES iteration for compressible case with $E = 10^7$ and $\nu = 0.3$

For a one-level preconditioner M, the condition number is

$$\kappa(M^{-1}A) \leq C \frac{1}{H^2} \left(1 + \frac{H}{\delta}\right)$$

where :

- δ size of the overlap
- H size of subdomain.

Why One-Level methods are not so good?

		AS	RAS	ORAS	SORAS
Nbr DOFs	Nbr subdom	iteration	iteration	iteration	iteration
66703	16	140	140	96	57
130433	32	245	216	146	109
266812	64	383	312	245	203
541838	128	566	460	480	398

Table: 2D Elasticity: number of of GMRES iteration for compressible case with $E = 10^7$ and $\nu = 0.3$

For a one-level preconditioner M, the condition number is

$$\kappa(M^{-1}A) \leq C \frac{1}{H^2} \left(1 + \frac{H}{\delta}\right)$$

where :

- δ size of the overlap
- H size of subdomain.

Two Level Domain Decomposition Methods

Given $V_H := \mathbf{span} Z$ an additive space. Z a set of vectors.

Define $R_H = Z^\intercal$ and $E := R_H A R_H^\intercal$ where E is much smaller than A

Enrich the one level preconditioner with Z (ie $Q = R_H^\intercal E^{-1} R_H$) :

- $M_{2,AD}^{-1} := Q + M^{-1}$.
- $M_{2,A-DEF_1}^{-1} := M^{-1}(I - AQ) + Q$.
- $M_{2,A-DEF_2}^{-1} := (I - QA)M^{-1} + Q$.
- $M_{2,BNN}^{-1} := (I - QA)M^{-1}(I - AQ) + Q$.

But what does the space V_H contain?

Two Level Domain Decomposition Methods

Given $V_H := \mathbf{span} Z$ an additive space. Z a set of vectors.

Define $R_H = Z^\intercal$ and $E := R_H A R_H^\intercal$ where E is much smaller than A

Enrich the one level preconditioner with Z (ie $Q = R_H^\intercal E^{-1} R_H$) :

- $M_{2,AD}^{-1} := Q + M^{-1}$.
- $M_{2,A-DEF_1}^{-1} := M^{-1}(I - AQ) + Q$.
- $M_{2,A-DEF_2}^{-1} := (I - QA)M^{-1} + Q$.
- $M_{2,BNN}^{-1} := (I - QA)M^{-1}(I - AQ) + Q$.

But what does the space V_H contain?

Two Level Domain Decomposition Methods

Given $V_H := \mathbf{span} Z$ an additive space. Z a set of vectors.

Define $R_H = Z^\intercal$ and $E := R_H A R_H^\intercal$ where E is much smaller than A

Enrich the one level preconditioner with Z (ie $Q = R_H^\intercal E^{-1} R_H$) :

- $M_{2,AD}^{-1} := Q + M^{-1}$.
- $M_{2,A-DEF_1}^{-1} := M^{-1}(I - AQ) + Q$.
- $M_{2,A-DEF_2}^{-1} := (I - QA)M^{-1} + Q$.
- $M_{2,BNN}^{-1} := (I - QA)M^{-1}(I - AQ) + Q$.

But what does the space V_H contain?

Two Level Domain Decomposition Methods

Given $V_H := \mathbf{span} Z$ an additive space. Z a set of vectors.

Define $R_H = Z^\intercal$ and $E := R_H A R_H^\intercal$ where E is much smaller than A

Enrich the one level preconditioner with Z (ie $Q = R_H^\intercal E^{-1} R_H$) :

- $M_{2,AD}^{-1} := Q + M^{-1}$.
- $M_{2,A-DEF_1}^{-1} := M^{-1}(I - AQ) + Q$.
- $M_{2,A-DEF_2}^{-1} := (I - QA)M^{-1} + Q$.
- $M_{2,BNN}^{-1} := (I - QA)M^{-1}(I - AQ) + Q$.

But what does the space V_H contain?

Two Level Domain Decomposition Methods

Given $V_H := \mathbf{span} Z$ an additive space. Z a set of vectors.

Define $R_H = Z^\intercal$ and $E := R_H A R_H^\intercal$ where E is much smaller than A

Enrich the one level preconditioner with Z (ie $Q = R_H^\intercal E^{-1} R_H$) :

- $M_{2,AD}^{-1} := Q + M^{-1}$.
- $M_{2,A-DEF_1}^{-1} := M^{-1}(I - AQ) + Q$.
- $M_{2,A-DEF_2}^{-1} := (I - QA)M^{-1} + Q$.
- $M_{2,BNN}^{-1} := (I - QA)M^{-1}(I - AQ) + Q$.

But what does the space V_H contain?

Two Level Domain Decomposition Methods

Given $V_H := \mathbf{span} Z$ an additive space. Z a set of vectors.

Define $R_H = Z^\intercal$ and $E := R_H A R_H^\intercal$ where E is much smaller than A

Enrich the one level preconditioner with Z (ie $Q = R_H^\intercal E^{-1} R_H$) :

- $M_{2,AD}^{-1} := Q + M^{-1}$.
- $M_{2,A-DEF_1}^{-1} := M^{-1}(I - AQ) + Q$.
- $M_{2,A-DEF_2}^{-1} := (I - QA)M^{-1} + Q$.
- $M_{2,BNN}^{-1} := (I - QA)M^{-1}(I - AQ) + Q$.

But what does the space V_H contain?

GenEO I approach to construct V_H

Find the eigenpairs $(\lambda_{i,k}, U_{i,k})$

$$A_i^N U_{i,k} = \lambda_{i,k} D_i R_{i,0}^T R_{i,0} A_i^N R_{i,0}^T R_{i,0} D_i U_{i,k}$$

Solved by ARPACK (Concurrently)

With :

- A_i^N the local unassembled matrix, resulting from the local bilinear form, with Neumann boundary condition on the interfaces.
- $R_{i,0} : \Omega_i \longrightarrow \Omega_i^\circ = \bigcup_{j \neq i} (\Omega_i \bigcap \Omega_j)$ a restriction operator.

Choose a number of eigenmodes τ_i for each subdomain then define

$$W_i = \left[D_i U_{i,1}, D_i U_{i,2}, \ldots, D_i U_{i,\tau_i} \right].$$

Finally

$$Z = [W_1, W_2, \ldots, W_N]$$

GenEO I approach to construct V_H

Find the eigenpairs $(\lambda_{i,k}, U_{i,k})$

$$A_i^N U_{i,k} = \lambda_{i,k} D_i R_{i,0}^T R_{i,0} A_i^N R_{i,0}^T R_{i,0} D_i U_{i,k}$$

Solved by ARPACK (Concurrently)

With :

- A_i^N the local unassembled matrix, resulting from the local bilinear form, with Neumann boundary condition on the interfaces.
- $R_{i,0} : \Omega_i \longrightarrow \Omega_i^\circ = \bigcup_{j \neq i} (\Omega_i \bigcap \Omega_j)$ a restriction operator.

Choose a number of eigenmodes τ_i for each subdomain then define

$$W_i = \left[D_i U_{i,1}, D_i U_{i,2}, \ldots, D_i U_{i,\tau_i} \right].$$

Finally

$$Z = [W_1, W_2, \ldots, W_N]$$

GenEO I approach to construct V_H

Find the eigenpairs $(\lambda_{i,k}, U_{i,k})$

$$A_i^N U_{i,k} = \lambda_{i,k} D_i R_{i,0}^T R_{i,0} A_i^N R_{i,0}^T R_{i,0} D_i U_{i,k}$$

Solved by ARPACK (Concurrently)

With :

- A_i^N the local unassembled matrix, resulting from the local bilinear form, with Neumann boundary condition on the interfaces.
- $R_{i,0} : \Omega_i \longrightarrow \Omega_i^\circ = \bigcup_{j \neq i} (\Omega_i \bigcap \Omega_j)$ a restriction operator.

Choose a number of eigenmodes τ_i for each subdomain then define

$$W_i = \left[D_i U_{i,1}, D_i U_{i,2}, \ldots, D_i U_{i,\tau_i} \right].$$

Finally

$$Z = [W_1, W_2, \ldots, W_N]$$

GenEO I approach to construct V_H

Find the eigenpairs $(\lambda_{i,k}, U_{i,k})$

$$A_i^N U_{i,k} = \lambda_{i,k} D_i R_{i,0}^T R_{i,0} A_i^N R_{i,0}^T R_{i,0} D_i U_{i,k}$$

Solved by ARPACK (Concurrently)

With :

- A_i^N the local unassembled matrix, resulting from the local bilinear form, with Neumann boundary condition on the interfaces.
- $R_{i,0} : \Omega_i \longrightarrow \Omega_i^\circ = \bigcup_{j \neq i} (\Omega_i \bigcap \Omega_j)$ a restriction operator.

Choose a number of eigenmodes τ_i for each subdomain then define

$$W_i = \left[D_i U_{i,1}, D_i U_{i,2}, \ldots, D_i U_{i,\tau_i} \right].$$

Finally

$$Z = [W_1, W_2, \ldots, W_N]$$

Theoretical result for GenEO I

Theorem (Spillane, Dolean, Hauret, Nataf, Pechstein, Scheichl)

If for all j: $0 < \lambda_{j,m_{j+1}} < \infty$*:*

$$\kappa(M_{AS,2}^{-1}A) \leq (1+k_0)\left[2+k_0\,(2k_0+1)\left(1+\tau\right)\right]$$

where :

- k_0 the maximum multiplicity of the interaction between subdomains.
- Parameter τ can be chosen arbitrarily small at the expense of a large coarse space.

SORAS-GenEO II approach to construct V_H

Find the eigenpairs $(\lambda_{i,k}, U_{i,k})$

$$A_i^N U_{i,k} = \lambda_{i,k} B_i U_{i,k}$$

Solved by ARPACK (Concurrently)

Find the eigenpairs $(\mu_{i,k}, V_{i,k})$

$$D_i B_i D_i V_{i,k} = \lambda_{i,k} A_i V_{i,k}$$

where :

- B_i the local unassembled matrix, resulting from the local bilinear form with Optimized interface conditions

Choose τ_i and γ_i for each subdomain then define

$$W_i = [D_i U_{i,1} \quad D_i U_{i,2} \quad \dots D_i U_{i,\tau_i}] \text{ And } H_i = [D_i V_{i,1} \quad D_i V_{i,2} \dots D_i V_{i,\gamma_i}]$$

$$Z^\tau = [W_1 \quad W_2 \dots \quad W_N] \text{ And } Z^\gamma = [H_1 \quad H_2 \quad \dots \quad H_N]$$

SORAS-GenEO II approach to construct V_H

Find the eigenpairs $(\lambda_{i,k}, U_{i,k})$

$$A_i^N U_{i,k} = \lambda_{i,k} B_i U_{i,k}$$

Find the eigenpairs $(\mu_{i,k}, V_{i,k})$

$$D_i B_i D_i V_{i,k} = \lambda_{i,k} A_i V_{i,k}$$

Solved by ARPACK (Concurrently)

where :

- B_i the local unassembled matrix, resulting from the local bilinear form with Optimized interface conditions

Choose τ_i and γ_i for each subdomain then define

$$W_i = [D_i U_{i,1} \quad D_i U_{i,2} \quad \dots D_i U_{i,\tau_i}] \text{ And } H_i = [D_i V_{i,1} \quad D_i V_{i,2} \dots D_i V_{i,\gamma_i}]$$

$$Z^\tau = [W_1 \quad W_2 \dots \quad W_N] \text{ And } Z^\gamma = [H_1 \quad H_2 \quad \dots \quad H_N]$$

SORAS-GenEO II approach to construct V_H

Find the eigenpairs $(\lambda_{i,k}, U_{i,k})$

$$A_i^N U_{i,k} = \lambda_{i,k} B_i U_{i,k}$$

Find the eigenpairs $(\mu_{i,k}, V_{i,k})$

$$D_i B_i D_i V_{i,k} = \lambda_{i,k} A_i V_{i,k}$$

Solved by ARPACK (Concurrently)

where :

- B_i the local unassembled matrix, resulting from the local bilinear form with Optimized interface conditions

Choose τ_i and γ_i for each subdomain then define

$W_i = [D_i U_{i,1} \quad D_i U_{i,2} \quad \dots D_i U_{i,\tau_i}]$ And $H_i = [D_i V_{i,1} \quad D_i V_{i,2} \dots D_i V_{i,\gamma_i}]$

$Z^\tau = [W_1 \quad W_2 \dots \quad W_N]$ And $Z^\gamma = [H_1 \quad H_2 \quad \dots \quad H_N]$

SORAS-GenEO II approach to construct V_H

Find the eigenpairs $(\lambda_{i,k}, U_{i,k})$

$$A_i^N U_{i,k} = \lambda_{i,k} B_i U_{i,k}$$

Find the eigenpairs $(\mu_{i,k}, V_{i,k})$

$$D_i B_i D_i V_{i,k} = \lambda_{i,k} A_i V_{i,k}$$

Solved by ARPACK (Concurrently)

where :

- B_i the local unassembled matrix, resulting from the local bilinear form with Optimized interface conditions

Choose τ_i and γ_i for each subdomain then define

$$W_i = \begin{bmatrix} D_i U_{i,1} & D_i U_{i,2} & \dots D_i U_{i,\tau_i} \end{bmatrix} \text{ And } H_i = \begin{bmatrix} D_i V_{i,1} & D_i V_{i,2} \dots D_i V_{i,\gamma_i} \end{bmatrix}$$

$$Z^\tau = [W_1 \quad W_2 \dots \quad W_N] \text{ And } Z^\gamma = [H_1 \quad H_2 \quad \dots \quad H_N]$$

SORAS-GenEO II approach to construct V_H

Find the eigenpairs $(\lambda_{i,k}, U_{i,k})$

$$A_i^N U_{i,k} = \lambda_{i,k} B_i U_{i,k}$$

Find the eigenpairs $(\mu_{i,k}, V_{i,k})$

$$D_i B_i D_i V_{i,k} = \lambda_{i,k} A_i V_{i,k}$$

Solved by ARPACK (Concurrently)

where :

- B_i the local unassembled matrix, resulting from the local bilinear form with Optimized interface conditions

Choose τ_i and γ_i for each subdomain then define

$$W_i = \begin{bmatrix} D_i U_{i,1} & D_i U_{i,2} & \dots D_i U_{i,\tau_i} \end{bmatrix} \text{ And } H_i = \begin{bmatrix} D_i V_{i,1} & D_i V_{i,2} \dots D_i V_{i,\gamma_i} \end{bmatrix}$$

$$Z^\tau = [W_1 \quad W_2 \dots \quad W_N] \text{ And } Z^\gamma = [H_1 \quad H_2 \quad \dots \quad H_N]$$

SORAS-GenEO II approach to construct V_H

Find the eigenpairs $(\lambda_{i,k}, U_{i,k})$

$$A_i^N U_{i,k} = \lambda_{i,k} B_i U_{i,k}$$

Find the eigenpairs $(\mu_{i,k}, V_{i,k})$

$$D_i B_i D_i V_{i,k} = \lambda_{i,k} A_i V_{i,k}$$

Solved by ARPACK (Concurrently)

where :

- B_i the local unassembled matrix, resulting from the local bilinear form with Optimized interface conditions

Choose τ_i and γ_i for each subdomain then define

$$W_i = \begin{bmatrix} D_i U_{i,1} & D_i U_{i,2} & \dots D_i U_{i,\tau_i} \end{bmatrix} \text{ And } H_i = \begin{bmatrix} D_i V_{i,1} & D_i V_{i,2} \dots D_i V_{i,\gamma_i} \end{bmatrix}$$

$$Z^\tau = [W_1 \quad W_2 \dots \quad W_N] \text{ And } Z^\gamma = [H_1 \quad H_2 \quad \dots \quad H_N]$$

SORAS-GenEO II approach to construct V_H

Find the eigenpairs $(\lambda_{i,k}, U_{i,k})$

$$A_i^N U_{i,k} = \lambda_{i,k} B_i U_{i,k}$$

Find the eigenpairs $(\mu_{i,k}, V_{i,k})$

$$D_i B_i D_i V_{i,k} = \lambda_{i,k} A_i V_{i,k}$$

Solved by ARPACK (Concurrently)

where :

- B_i the local unassembled matrix, resulting from the local bilinear form with Optimized interface conditions

Choose τ_i and γ_i for each subdomain then define

$$W_i = \begin{bmatrix} D_i U_{i,1} & D_i U_{i,2} & \dots D_i U_{i,\tau_i} \end{bmatrix} \text{ And } H_i = \begin{bmatrix} D_i V_{i,1} & D_i V_{i,2} \dots D_i V_{i,\gamma_i} \end{bmatrix}$$

$$Z^\tau = [W_1 \quad W_2 \dots \quad W_N] \text{ And } Z^\gamma = [H_1 \quad H_2 \quad \dots \quad H_N]$$

Two level SORAS-GENEO-2 preconditioner

Theorem (Haferssas, Jolivet and N., 2015)

Let γ and τ be user-defined targets. Then, the eigenvalues of the two-level SORAS-GenEO-2 preconditioned system satisfy the following estimate

$$\frac{1}{1+\frac{k_1}{\tau}} \leq \lambda(M_{SORAS,2}^{-1} A) \leq \max(1, k_0\,\gamma)$$

What if one level method is M_{OAS}^{-1}:

$$\text{Find } (\mathbf{V}_{jk}, \lambda_{jk}) \in \mathbb{R}^{\#\mathcal{N}_i} \setminus \{0\} \times \mathbb{R} \text{ such that}$$
$$A_i^{Neu} \mathbf{V}_{ik} = \lambda_{ik} D_i B_i D_i \mathbf{V}_{ik} \ .$$

Numerical results via a Domain Specific Language

`FreeFem++` (`http://www.freefem.org/ff++`), `F.Hecht`, with:

- Metis Karypis and Kumar 1998
- SCOTCH Chevalier and Pellegrini 2008
- UMFPACK Davis 2004
- ARPACK Lehoucq et al. 1998
- MPI Snir et al. 1995
- Intel MKL
- PARDISO Schenk et al. 2004
- MUMPS Amestoy et al. 1998
- PaStiX Hénon et al. 2005
- Slepc via PETSC

Runs on PC (Linux, OSX, Windows) and HPC (Babel@CNRS, HPC1@LJLL, Titane@CEA via GENCI PRACE)

Why use a DS(E)L instead of `C/C++/Fortran/..` ?

- performances close to low-level language implementation,
- hard to beat something as simple as:

$$\begin{aligned}\mathbf{varf}\ \mathrm{a}(\mathrm{u},\ \mathrm{v}) = \mathbf{int3d}(\mathrm{mesh})([\mathbf{dx}(\mathrm{u}),\ \mathbf{dy}(\mathrm{u}),\ \mathbf{dz}(\mathrm{u})]'\ *\ [\mathbf{dx}(\mathrm{v}),\ \mathbf{dy}(\mathrm{v}),\ \mathbf{dz}(\mathrm{v})])\\ +\ \mathbf{int3d}(\mathrm{mesh})(\mathrm{f}\ *\ \mathrm{v})\ +\ \mathbf{on}(\mathrm{boundary\_mesh})(\mathrm{u} = 0)\end{aligned}$$

Machine used for scaling tests

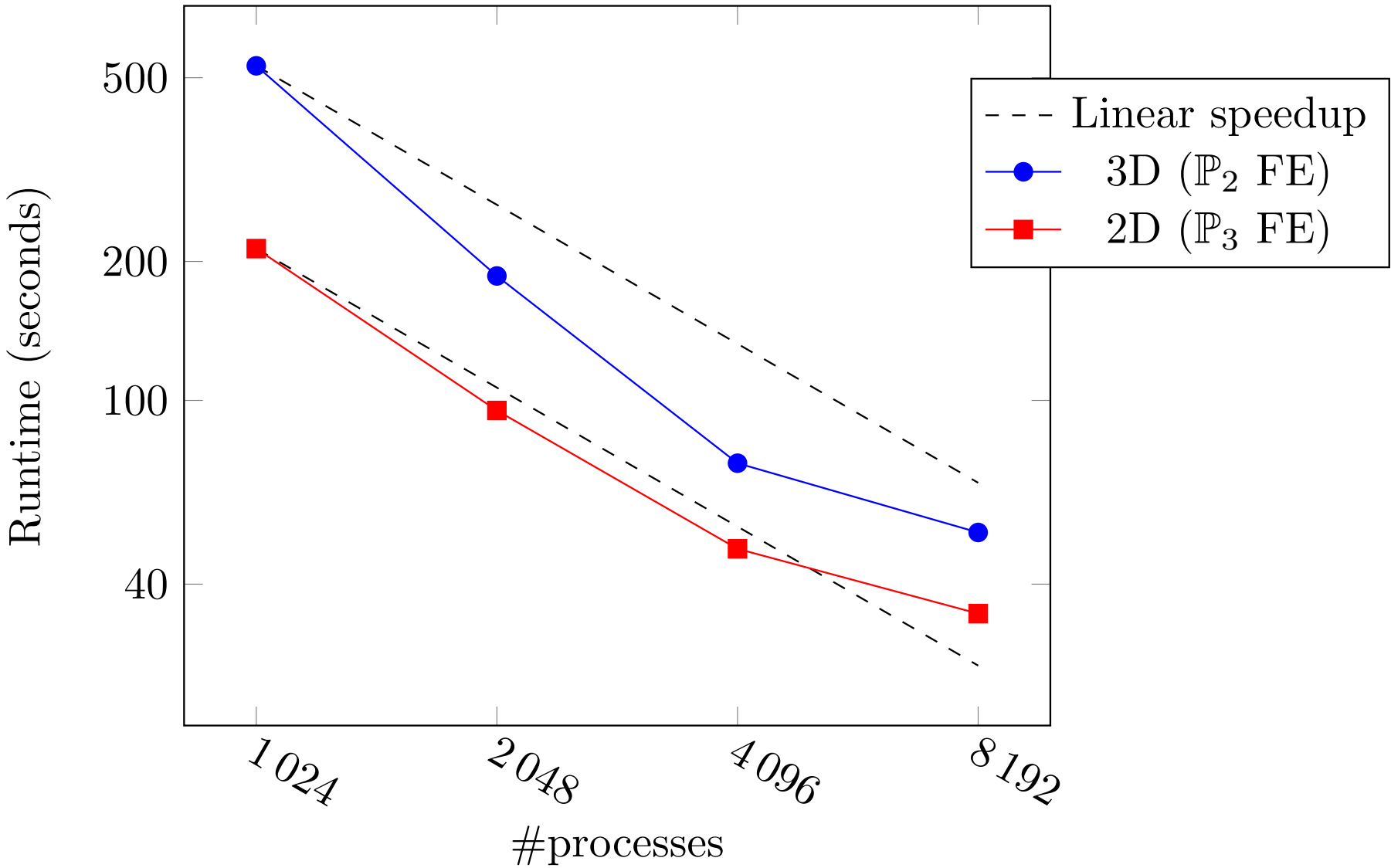
Curie Thin Nodes

- 5,040 compute nodes.
- 2 eight-core Intel Sandy Bridge@2.7 GHz per node.
- IB QDR full fat tree.
- 1.7 PFLOPs peak performance.

Jolivet, P., Hecht, F., Nataf, F., Prud'homme, C. *Scalable Domain Decomposition Preconditioners For Heterogeneous Elliptic Problems* Supercomputing 2013. Best paper finalist.

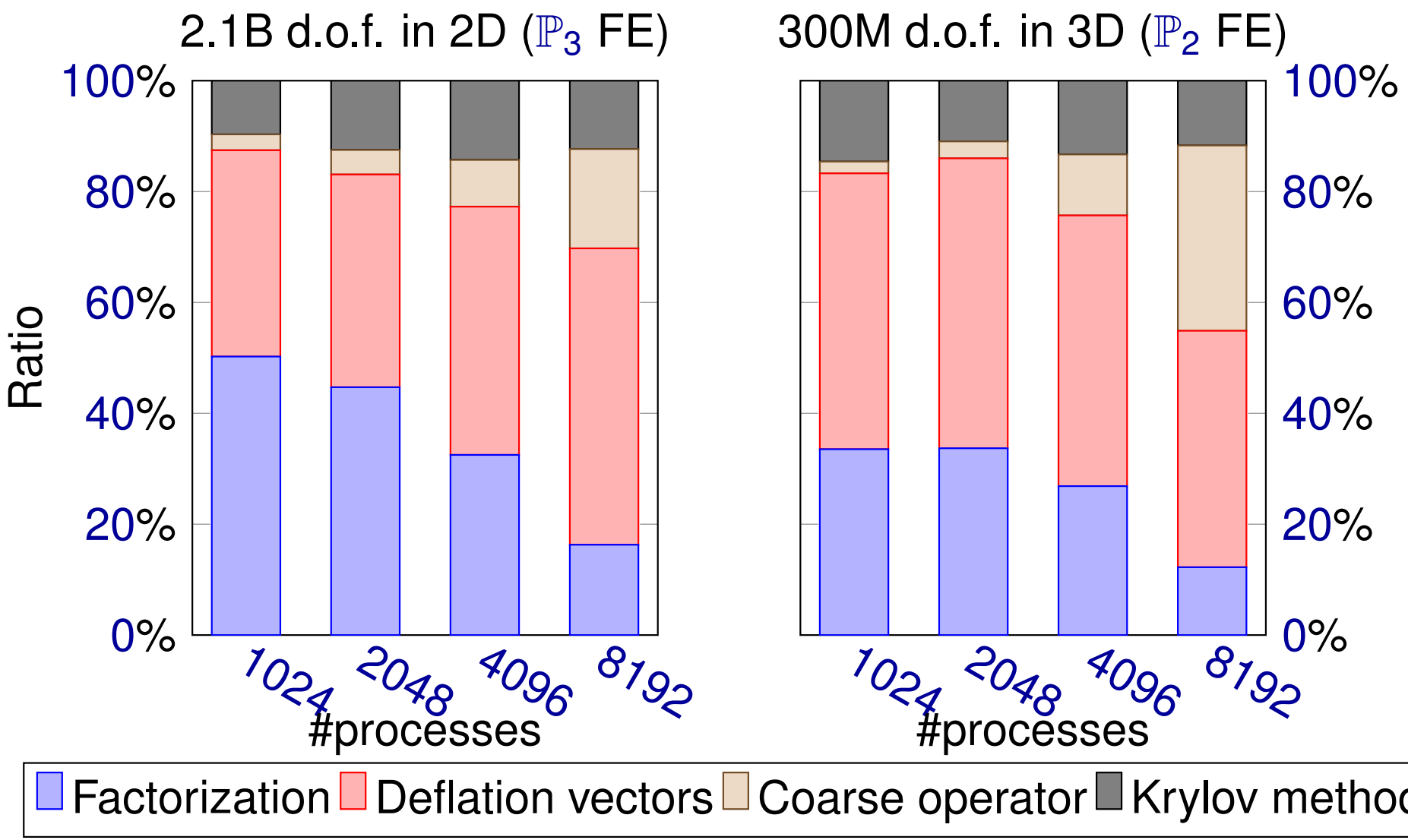
Strong scaling (linear elasticity)

1 subdomain/MPI process, 2 OpenMP threads/MPI process.



Strong scaling (linear elasticity)

1 subdomain/MPI process, 2 OpenMP threads/MPI process.



Highly Heterogeneous Coefficients

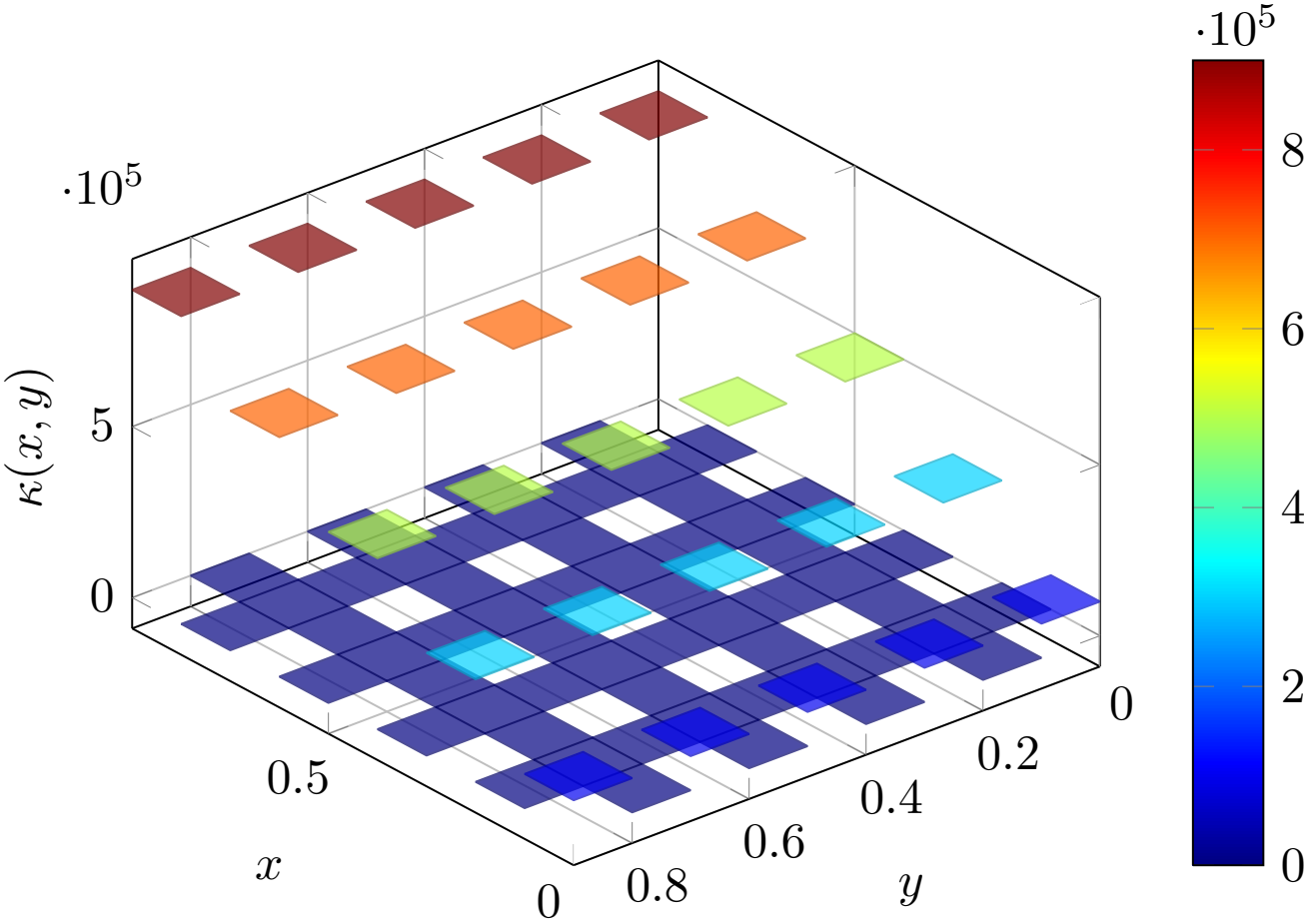
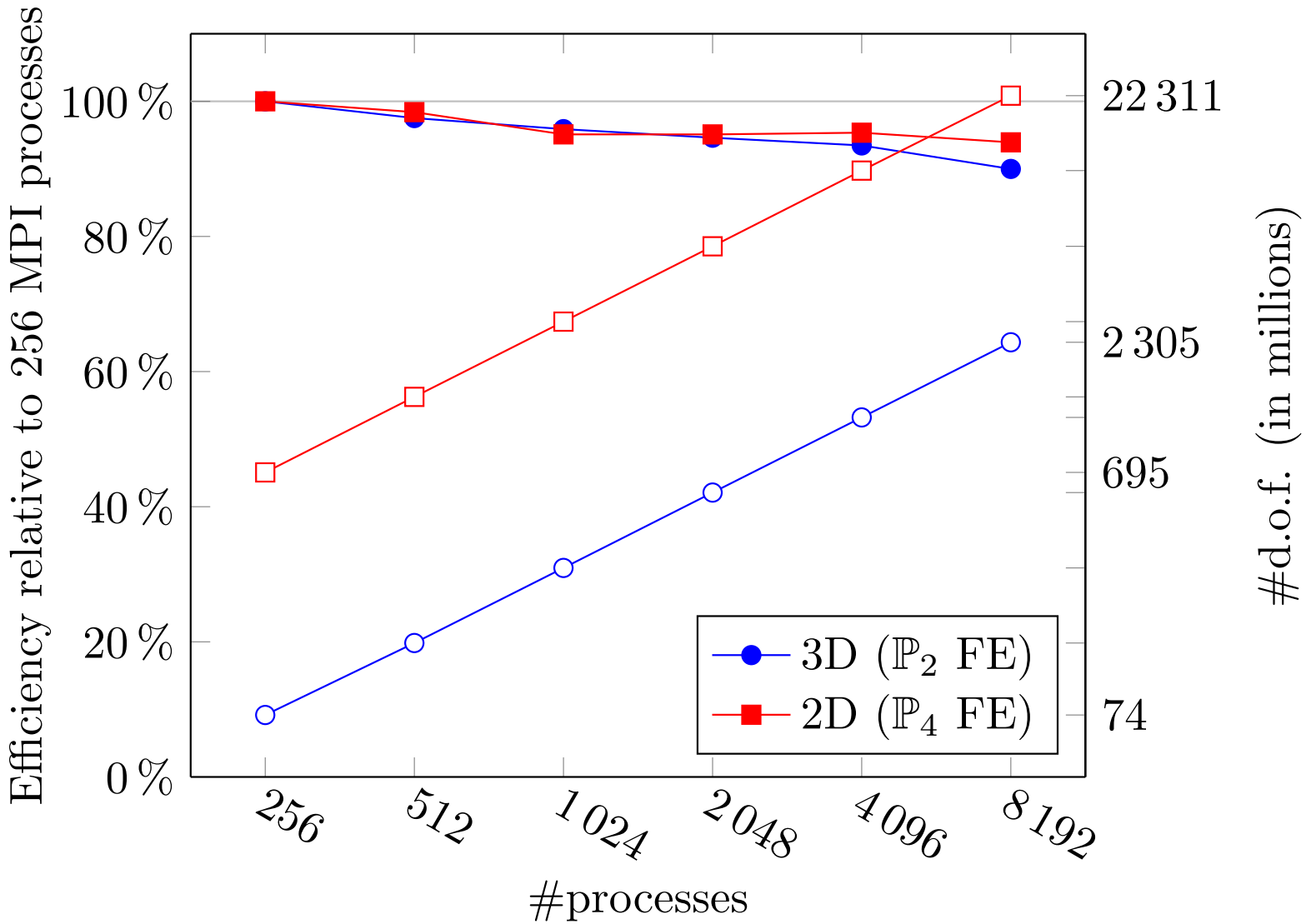


Figure: Two dimensional diffusivity κ

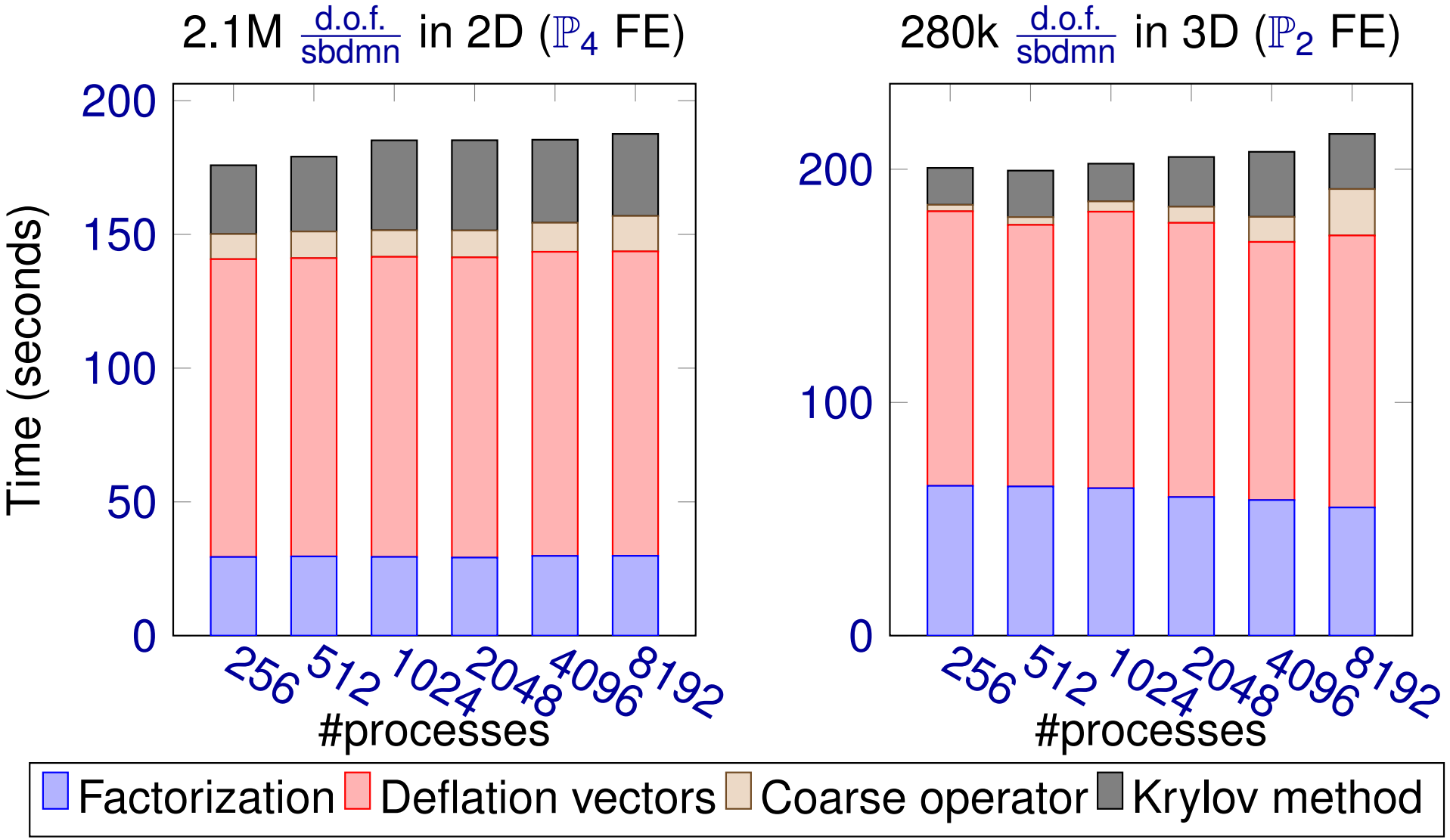
Weak scaling (scalar diffusion equation)

1 subdomain/MPI process, 2 OpenMP threads/MPI process.



Weak scaling (scalar diffusion equation)

1 subdomain/MPI process, 2 OpenMP threads/MPI process.



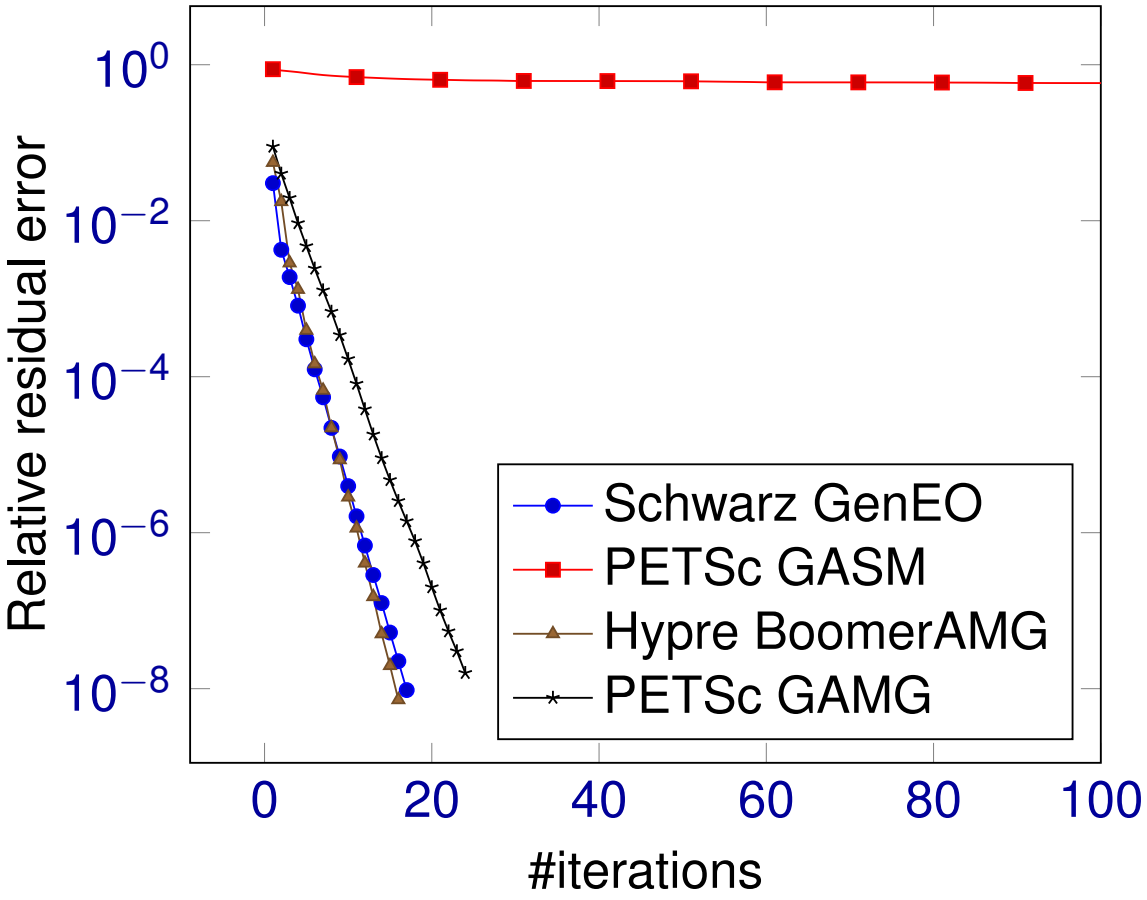
Comparison

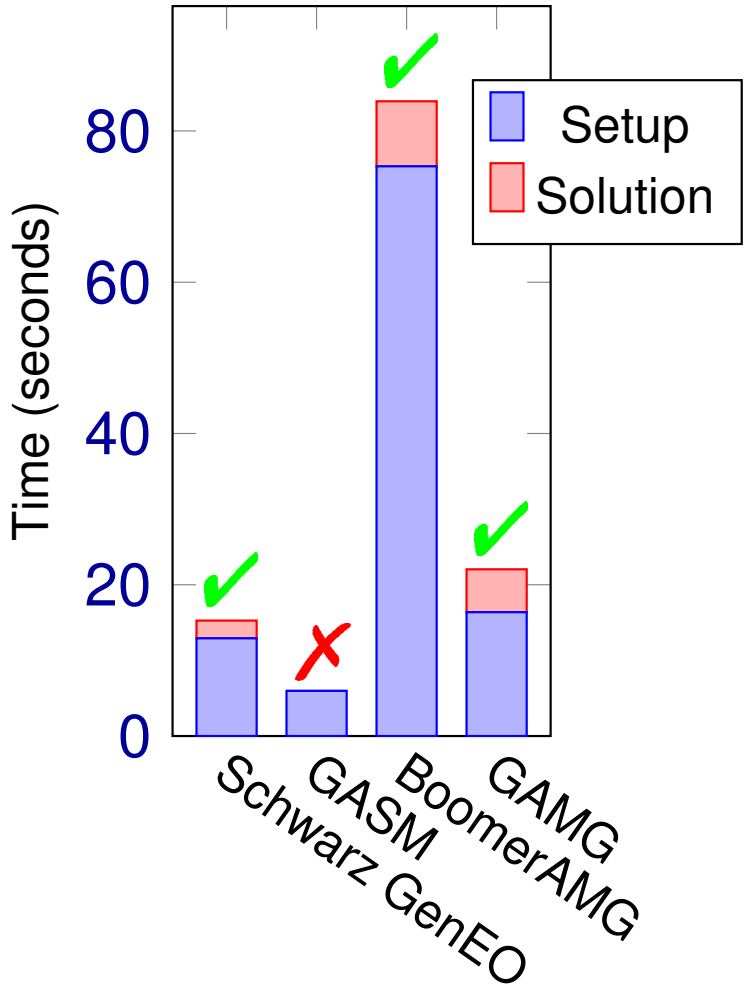
Comparing performance of setup and solution phases between our solver against purely algebraic (+ near null space) solvers:

- GASM – one-level domain decomposition method (ANL),
- Hypre BoomerAMG – algebraic multigrid (LLNL),
- GAMG – algebraic multrigrid (ANL/LBL).

Solution of a linear system I

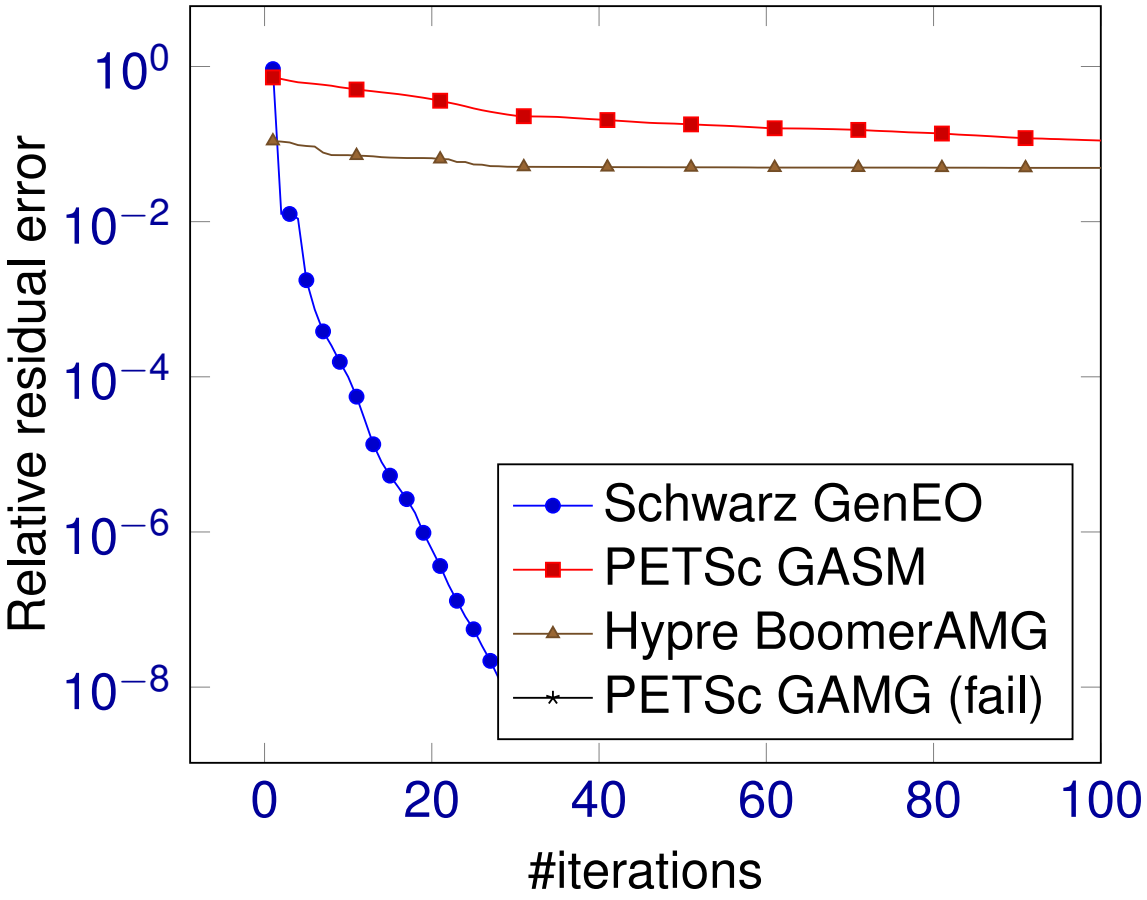
Homogeneous 3D Poisson equation discretized by $\mathbb{P}_1$ FE solved on 2,048 MPI processes, 111M d.o.f.

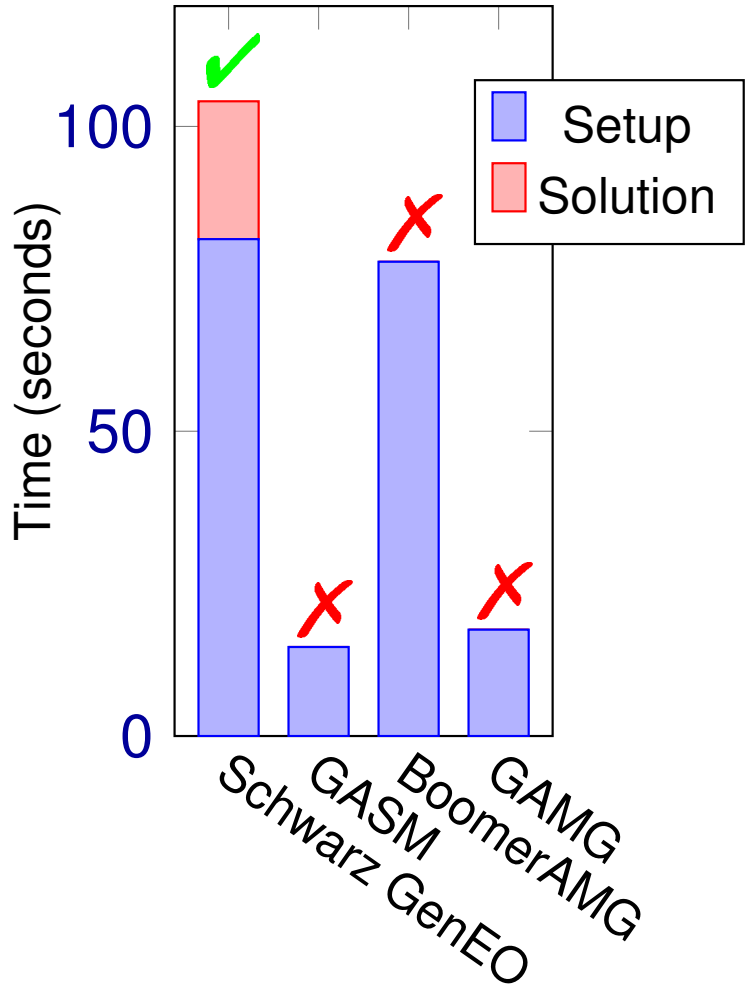




Solution of a linear system II

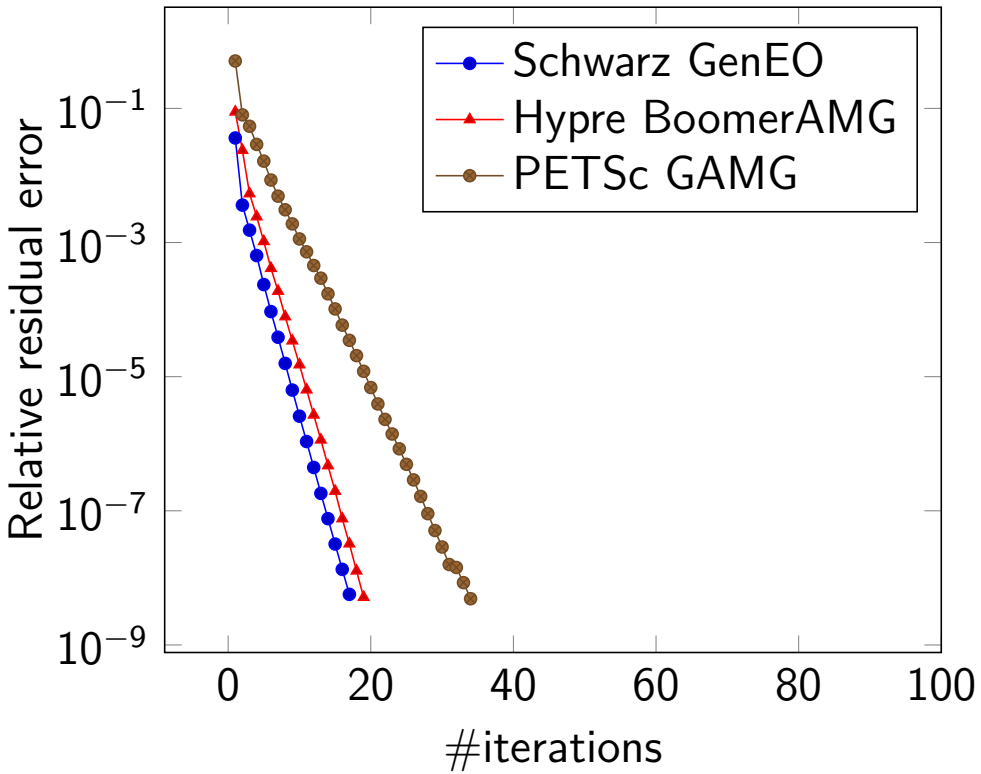
Heterogeneous 3D linear elasticity equation discretized by $\mathbb{P}_2$ FE solved on 4,096 MPI processes, 127M d.o.f.

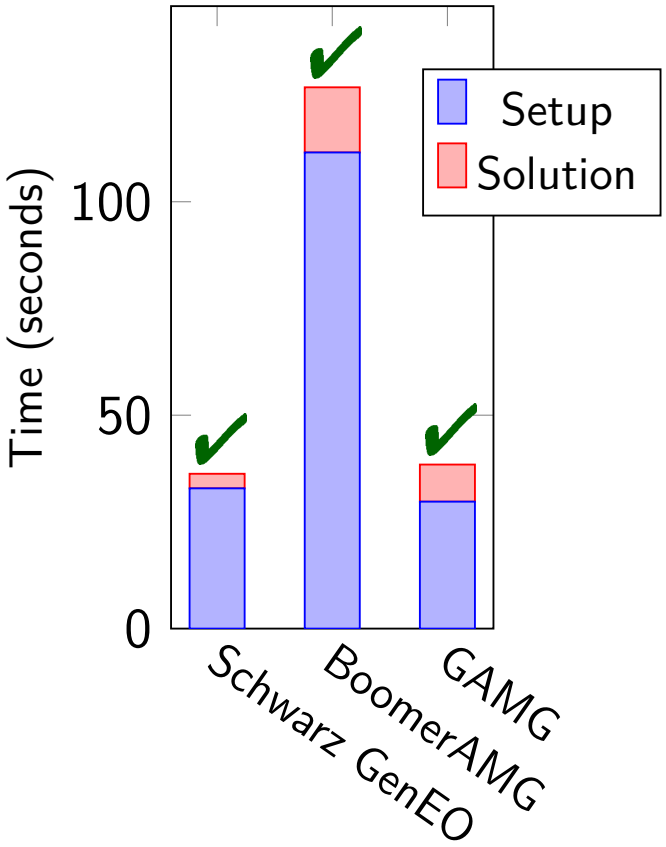




Solution of a linear system I

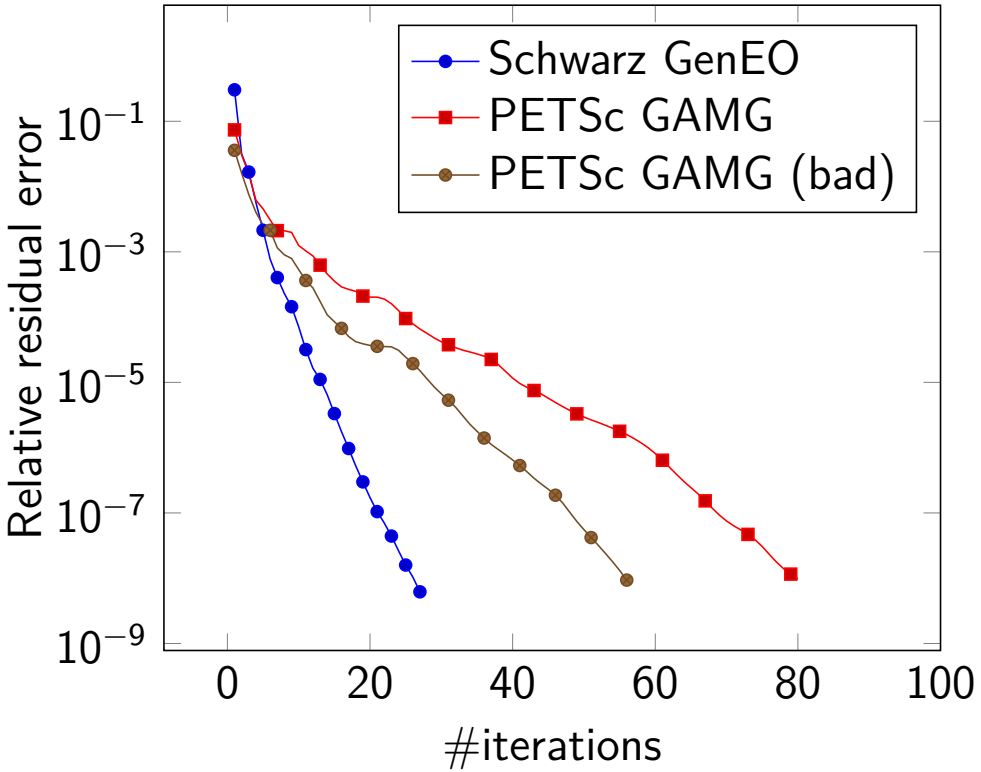
Homogeneous 3D Poisson equation discretized by $\mathbb{P}_1$ FE solved on 4,096 MPI processes, 217M d.o.f.

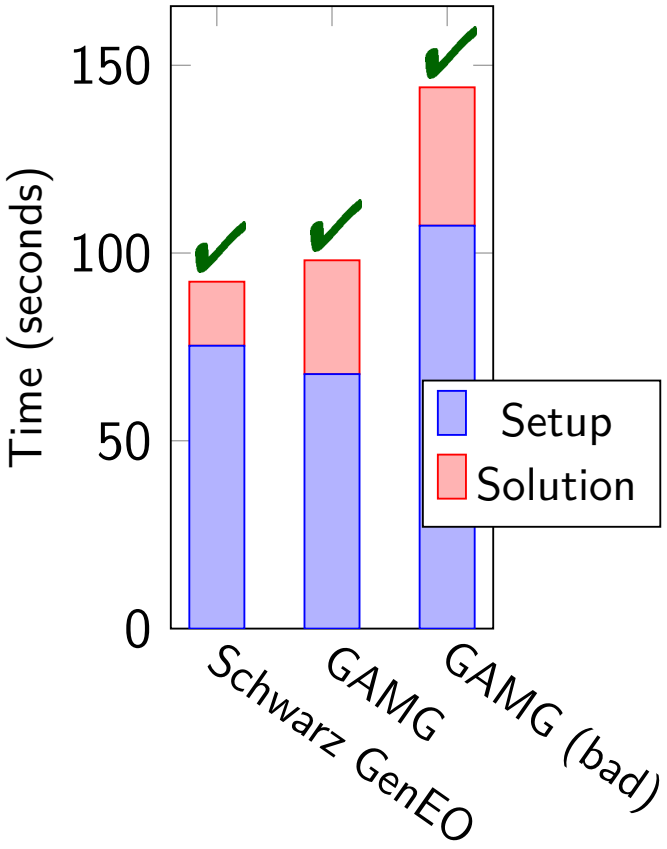




Solution of a linear system II

Heterogeneous 3D linear elasticity equation discretized by $\mathbb{P}_2$ FE solved on 4,096 MPI processes, 262M d.o.f.





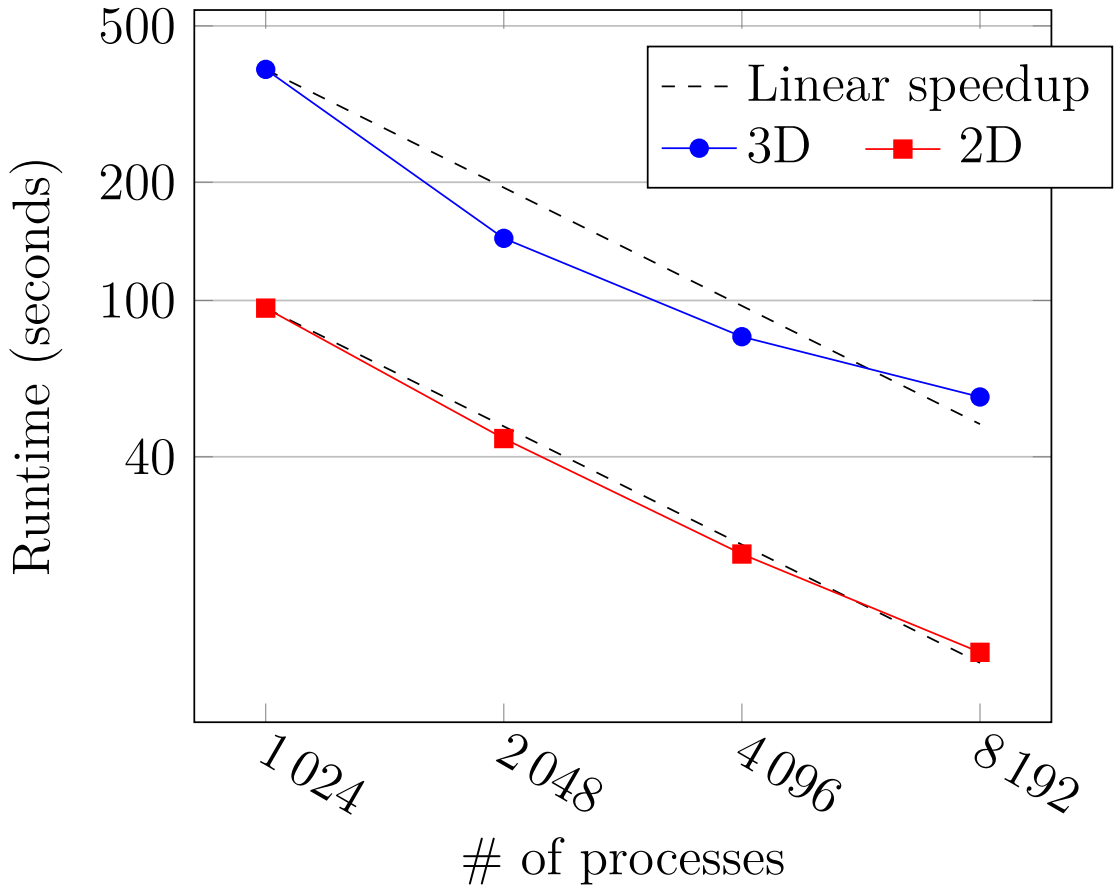
Machine used for scaling tests

Turing, IDRIS-Genci project

- IBM Blue Gene/Q
- 6144 compute nodes (16 core per node @1.6 GHz).
- 1.258 PFLOPs peak performance

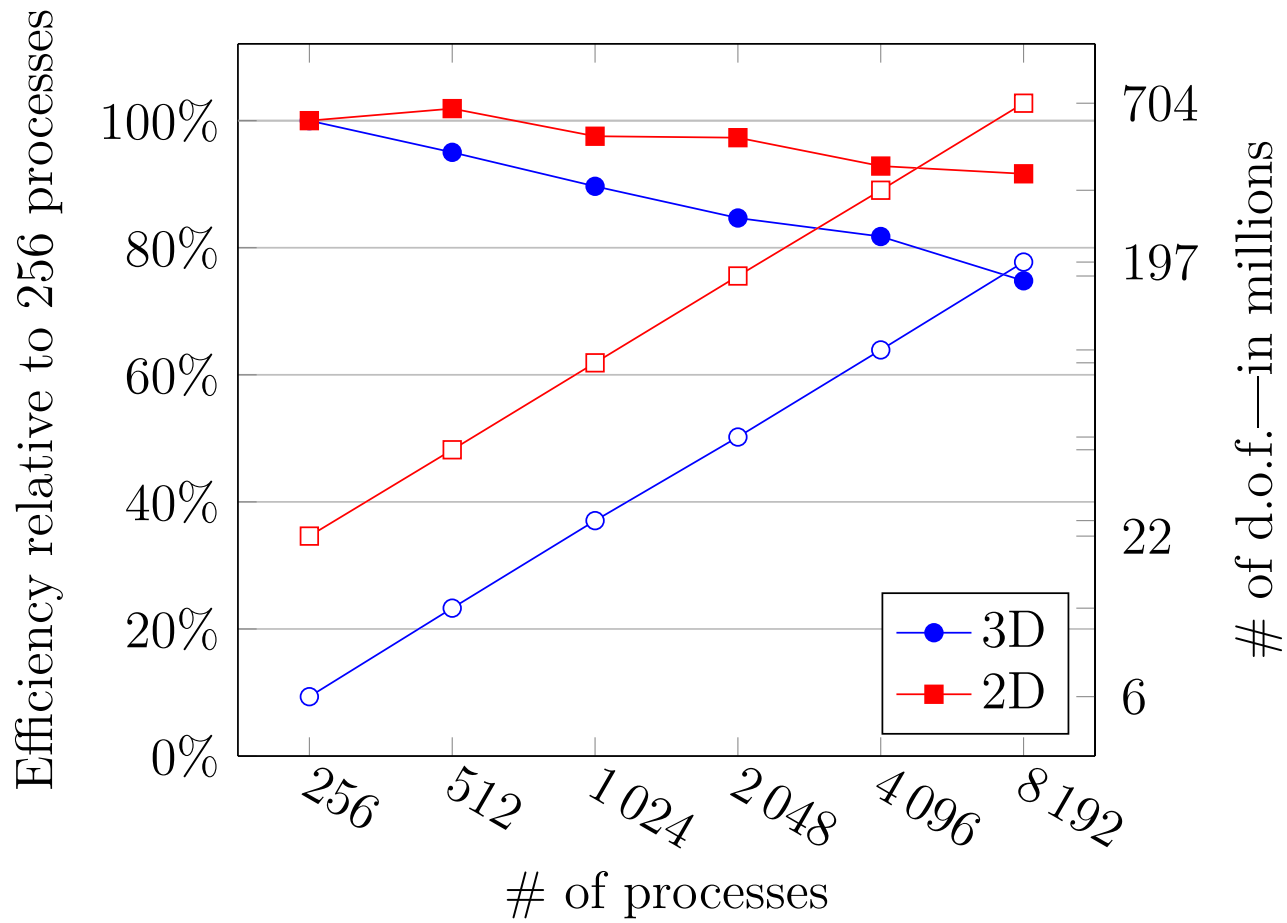
Strong scaling (Stokes problem), (with FreeFem++ and HPDDM)

Stokes problem with automatic mesh partition. Driven cavity
Discretized by $\mathbb{P}_2\backslash\mathbb{P}_1$ FE 50M d.o.f. (3D), 100M d.o.f. (2D)



Weak scaling (Nearly-Incompressible linear elasticity: Steel and Rubber), (with FreeFem++ and HPDDM)

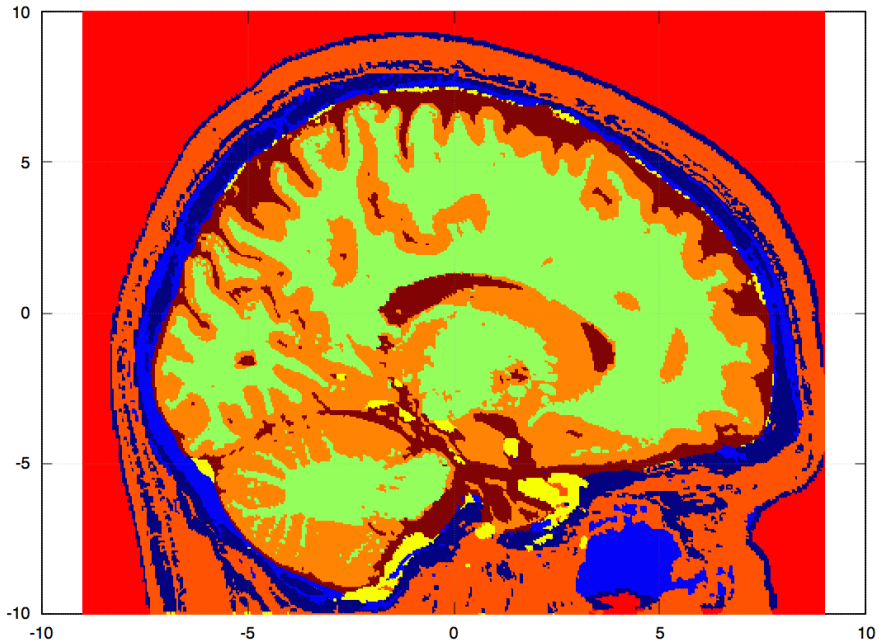
1 subdomain/MPI process, 2 OpenMP threads/MPI process.



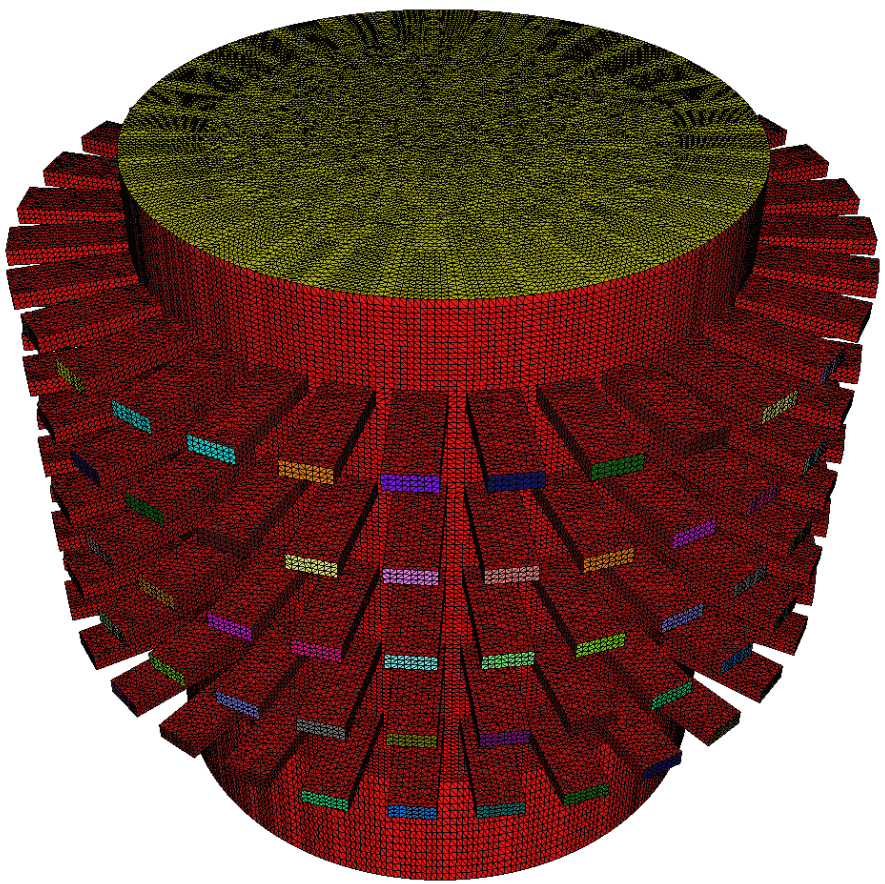
Altix UV 2000, Laboratory J.L. Lions or Institut du Calcul Scientifique, UPMC

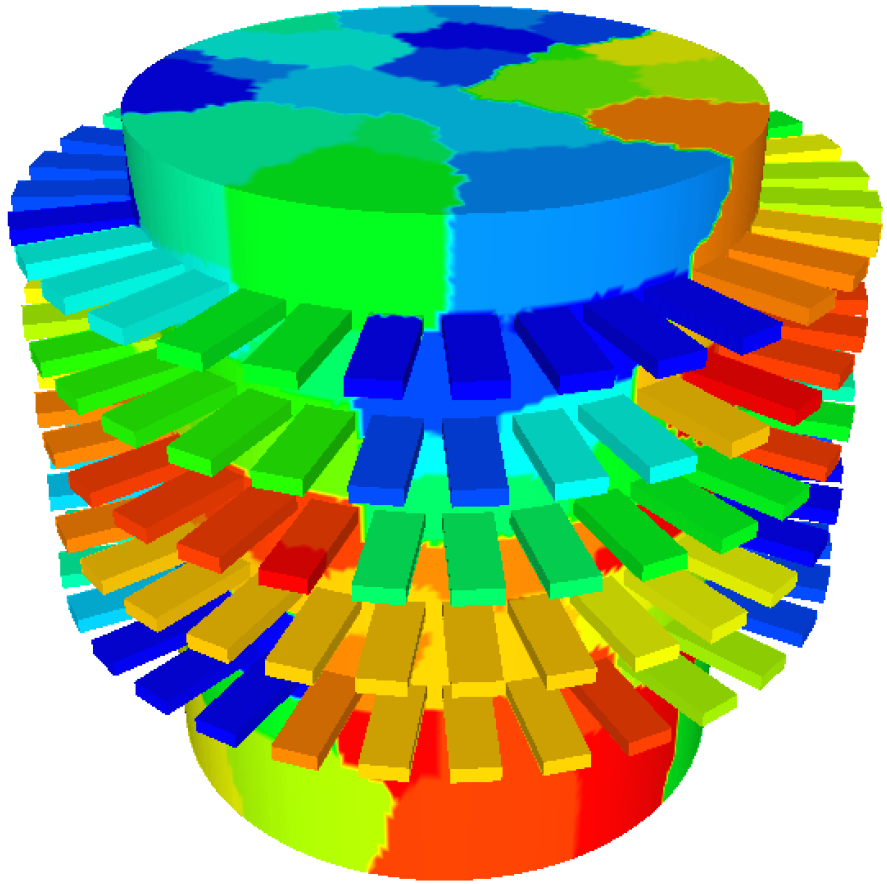
- SGI
- 160 cores or 1024 cores
- Intel Xeon 64 bits at 2.5GHz

- $f = 1GHz$, waveguides (ceramic)
- Maxwell system with a realistic Brain Model



Maxwell system for Brain Imaging

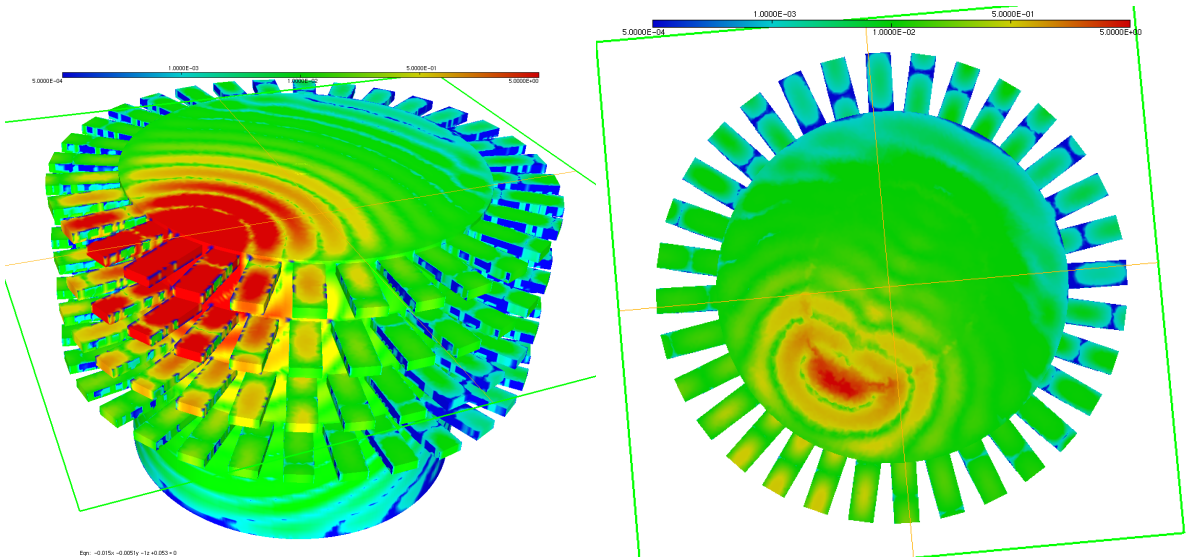




5 rings of 32 antennas each (FreeFem++, F. Hecht and P.H tournier)

SGI UV2000, Institut du Calcul Scientifique, UPMC

- 9.3M degrees of freedom
- Computations done using 128 processors
- 40s wall time for 1 right-hand side
- 500s wall time for 32 right-hand sides (16s per rhs)



Electric field with gel only, field scattered by the head

Schur complement method

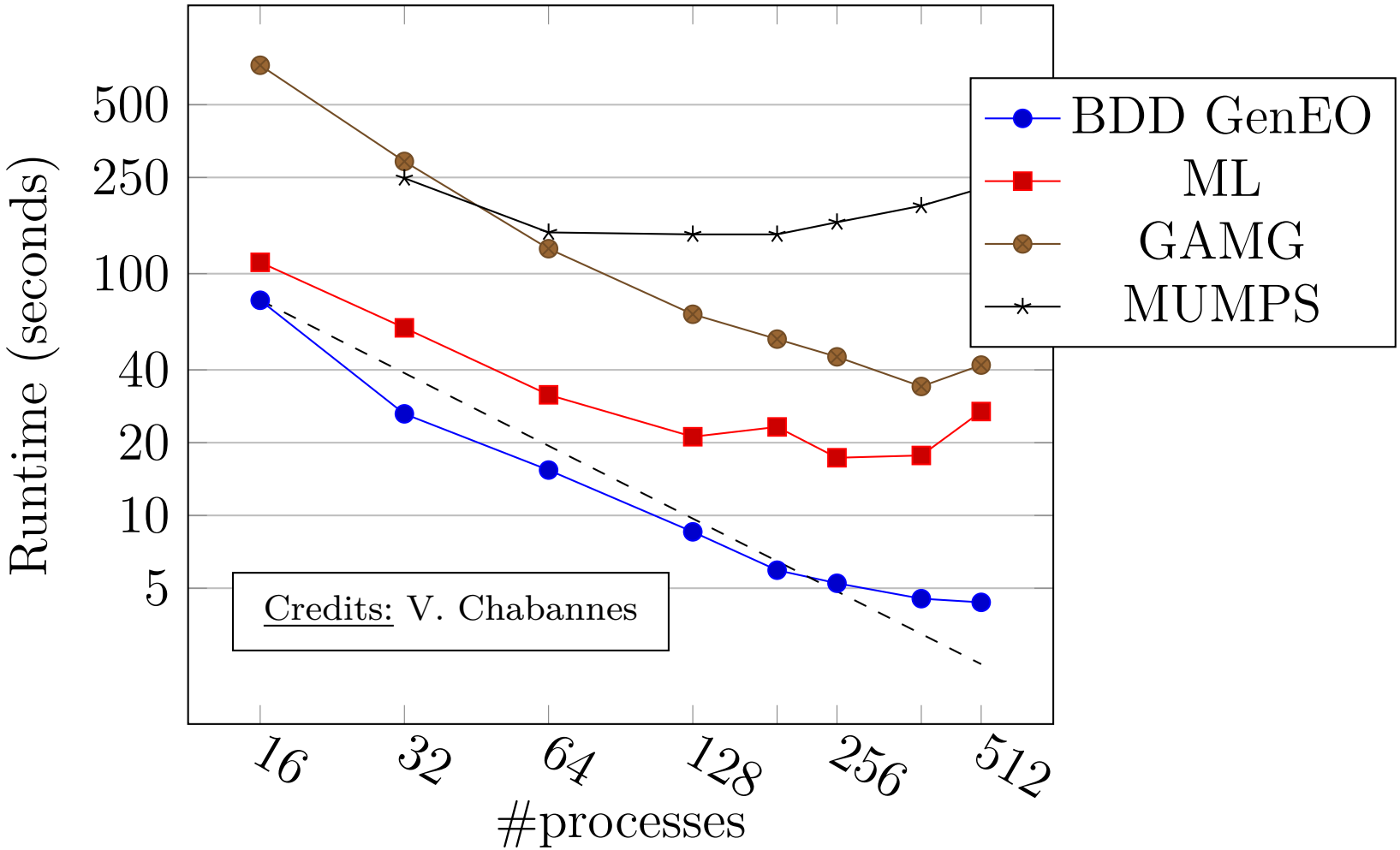


Fig 1: Homogeneous 2D elasticity discretized by $\mathbb{P}_3$ FE, 23M d.o.f.

Conclusion: HPDDM

Summary

- Using one (or two) generalized eigenvalue problems and projection preconditioning we are able to achieve a targeted convergence rate.
- Versatile solver for highly heterogeneous coefficients: Poisson and Darcy scalar equations, Stokes, elasticity (rubber and steel) and Maxwell systems
- Availability
  - public release of FreeFem++
  - interfaced with Feel++
  - as a stand alone library HPDDM C++/MPI library `https://github.com/hpddm`

Future

- Multilevel extension of the coarse operator,
- Nonlinear time dependent problem (Reuse of the coarse space)

Conclusion

Democratization of Scientific Simulations

- Dedicated High level language $\rightarrow$ FreeFem++
- Hide parallelism to end-users $\rightarrow$ HPDDM
- Cloud and/or middle size clusters

Links

- HPDDM library `https://github.com/hpddm/hpddm` (A small C code is provided as an example)
- FreeFem++ (v3.38) `http://www.freefem.org/ff++/`
- Lecture Notes : An Introduction to Domain Decomposition Methods: algorithms, theory and parallel implementation, V. Dolean, P. Jolivet and F. Nataf `https://hal.archives-ouvertes.fr/cel-01100932v3`

THANK YOU FOR YOUR ATTENTION!

Conclusion

Democratization of Scientific Simulations

- Dedicated High level language → FreeFem++
- Hide parallelism to end-users → HPDDM
- Cloud and/or middle size clusters

Links

- HPDDM library `https://github.com/hpddm/hpddm` (A small C code is provided as an example)
- FreeFem++ (v3.38) `http://www.freefem.org/ff++/`
- Lecture Notes : An Introduction to Domain Decomposition Methods: algorithms, theory and parallel implementation, V. Dolean, P. Jolivet and F. Nataf `https://hal.archives-ouvertes.fr/cel-01100932v3`

THANK YOU FOR YOUR ATTENTION!